Danny Goodman's Apple Guide Starter Kit

Danny Goodman
Jeremy Joan Hewes

Includes easy-to-use software for developing Apple Guide tutorials

DANNY GOODMAN'S

APPLE GUIDE STARTER KIT

Danny Goodman

and Jeremy Joan Hewes

Addison-Wesley Publishing Company

Reading, Massachusetts•Menlo Park, California•New York
Don Mills, Ontario•Wokingham, England•Amsterdam
Bonn•Sydney•Singapore•Tokyo•Madrid•San Juan
Paris•Seoul•Milan•Mexico City•Taipei

Library of Congress Cataloging-in-Publication Data

Goodman, Danny.
[Apple guide starter kit]
Danny Goodman's Apple guide starter kit / Danny Goodman and Jeremy Joan Hewes.
p. cm.
Includes index.
ISBN 0-201-48349-1
1. Macintosh (Computer)—Programming. 2. Computer software– Development. 3. Apple Guide. I. Hewes, Jeremy Joan. II. Title. III. Title: Apple guide starter kit.
QA76.8.M3G644 1995
005.265—dc20 95-6089
CIP

Sponsoring Editor: Martha Steffen
Project Manager: John Fuller
Production Coordinator: Ellen Savett
Cover design: Jean Seal
Text design: DeNee Reiton Skipper
Set in 11 point Meridien by DeNee Reiton Skipper

1 2 3 4 5 6 7 8 9 -MA- 9998979695
First printing, July 1995

Addison-Wesley books are available for bulk purchases by corporations, institutions, and other organizations. For more information, please contact the Corporate, Government, and Special Sales Department at (800) 238-9682.

CONTENTS

ACKNOWLEDGMENTS

Our thanks to everyone who helped us create this book and software kit. We received suggestions, encouragement, and technical assistance from many people, especially these dedicated and generous folks.

The Apple Guide team at Apple: Peter Commons, Sharon Everson, Brian Glenn, Bob Hagenau, Glenn Katz, Brian Lawley, Greg Maletic, Brian McGhie, John Powers, Georgianne Puckett, Alan Samuel, and Daphne Steck.

The Instructional Products group at Apple: Lissy Abraham, Jose Arcellana, Elizabeth Boling, Kay Burnett, Pat Chaney, Heather Furmidge, Shemin Gau, Stella Hackell, Pete Harbeson, Jeff Hellman, Steve Hiatt, Mimi Hills, Naomi Horiuchi, Vera Horiuchi, Channing Hughes, Freda Husic, Stef Jones, Sharon Joyner, Kevin Knabe, Kristy Knabe, Stephanie Koester, Laila Millar, James Miyake, Jim Palmer, Brett Sher, Melissa Sleeter, Andy Swartz, John Trumble, and Molly Tyson.

Our reviewers and software testers: David Case, Sue Espinosa, Ken Landau, Al Staley, Eli Verschleisser, and the wizards of guideWorks.

The Addison-Wesley team: John Fuller, Kaethin Prizer, Ellen Savett, and Martha Steffen.

And our indispensable support group: Betsy Dilernia, F. N. Doty, and Linda Racine.

This book and the kit's software benefited enormously from the contributions of the people named above. Any errors or inconsistencies are strictly our own.

PART I

QUICK START

ONE

OVERVIEW: APPLE GUIDE AND THE APPLE GUIDE STARTER KIT

In August 1994, Apple Computer introduced Apple Guide as part of its System 7.5 for the Macintosh. Apple Guide is a new component of the system software that delivers step-by-step on-screen instructions for using the Macintosh. This sophisticated online help replaces the majority of printed documentation for Macintosh computers. (The manual now contains only setup and expansion instructions, safety and maintenance information, and troubleshooting tips.)

Although online help isn't a new development, Apple Guide brings several unique features to this technology. First, Apple Guide's instructions are interactive—you can take the action described without having to close or hide the Guide window. Second, in most instances Apple Guide shows you where to take the action—by drawing a circle around the appropriate icon, menu title, or area in a window. Sometimes Apple Guide even opens a folder or a control panel for you.

Most of the time, Apple Guide also knows what action you've taken, which window is active, or what item your system is missing. If you're ahead of the steps in the instructions, it skips the ones you don't need to see. If you have the wrong window on top, it alerts you and reminds you of the correct action to take.

In short, Apple Guide is no ordinary read-the-text-close-the-window-try-to-remember help system. The "engine" for this elegant instructional technology is provided with every Macintosh, along with several databases that cover the system software. With the tools and information in the Apple Guide Starter Kit, you can write instructions and create Apple Guide databases for any task or procedure with the Macintosh.

The Apple Guide Starter Advantage

This book-and-disk combination is your starter kit for building all kinds of Apple Guide files. With the Apple Guide Starter program on the disk, you can make Guides quickly and easily. You don't have to learn a scripting language, write coded files, or use several different files and programs to produce your database (the procedure you'd have to follow if you didn't have this program).

Using Apple Guide Starter, you simply choose the options for your Guide, fill in the names of tasks and topics, and then write instructions for those tasks and Topics, all in an on-screen environment that looks very much like Apple Guide itself. The Apple Guide Starter program does the rest of the work for you—creating a text file with the correct scripts and syntax, putting your instructions in the proper format, and compiling all that information into a Guide.

Because the Apple Guide Starter program makes Guide authoring so simple, you can create custom databases for virtually any task or procedure. Some of the ways you can put the program to immediate use include making Guides for the following:

- Tasks you (or your coworkers or clients) perform infrequently, such as creating special-format documents or printing labels or envelopes
- Tasks that involve more than one program and thus would require a user to look in several places for complete instructions
- The "top ten" questions for your work environment—the ten tasks for which people need help most often (or complain about the most loudly)
- Any number of customized procedures for a business, such as preparing invoices, updating archives or inventories, connecting to an online information service, or searching a large database

- A variety of school uses, such as quizzes, instructions for classroom projects, computer lab assignments, and computer training for teachers and students
- Collections of tips for automating tasks and getting the most from complex software, such as a spreadsheet or page-layout program
- Procedures for keeping sensitive information secure and for backing up important data
- Reference sources, such as a set of explanations related to your business or a list of suggested features that others might use to increase their productivity and efficiency with the computer (or with a specific program)
- "Trial" Guides—small compiled databases that you can use to try out new instructional approaches or to get others' feedback before using the approach for a larger Guide file

Visual Vocabulary

To introduce Apple Guide and the Apple Guide Starter program, we've combined brief descriptions of their major components with pictures of their respective windows. Once you've planned and written some Guide content, these elements will be familiar, so use these figures for orientation until you begin working with Guide Starter.

Apple Guide and Apple Guide Starter

Figure 1-1 provides a visual quick reference for the Access window of an Apple Guide database (Macintosh Guide) and an Apple Guide Starter application.

The Access window opens when a user chooses a Guide from the Apple Guide menu (at the right-hand side of the menu bar, with the question mark–light bulb icon). This window provides users with the categories and terms that describe the content of a Guide database. The Access window can be in the Full format (shown in Figure 1-1) or in the Single-list format, which shows one list containing Headers and Topics. In the Full Access window, a user can view content in three ways: Topics, a category listing (see Figure 1-1); Index, a listing of keywords; or Look For, a lookup feature that searches for a word or phrase the user types.

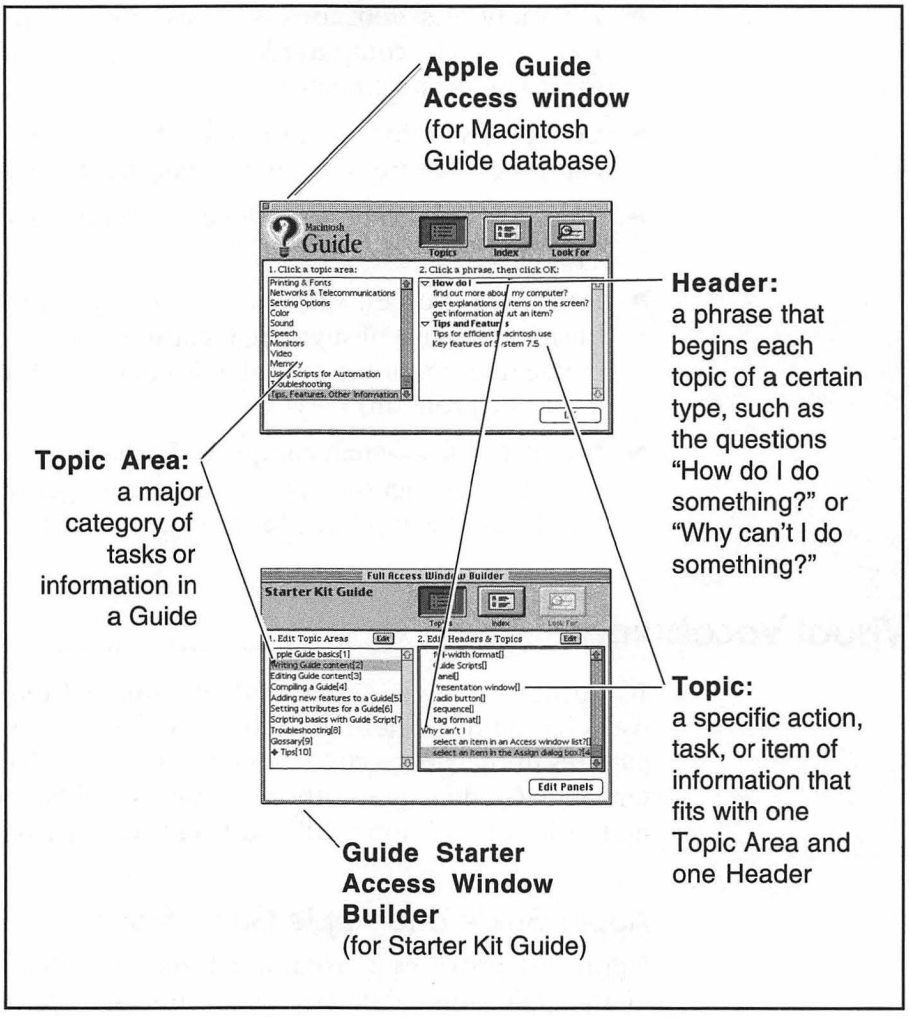


Figure 1-1
Common elements of Access window in a Guide and Apple Guide Starter application

Figure 1-2 shows the major elements of the Presentation window, which contains instructions for the topic a user selects in the Access window.

The Presentation window (also commonly called the panel window) delivers a series of steps or statements—the panels. On-screen events, such as a coach mark indicating a specific item or menu, are

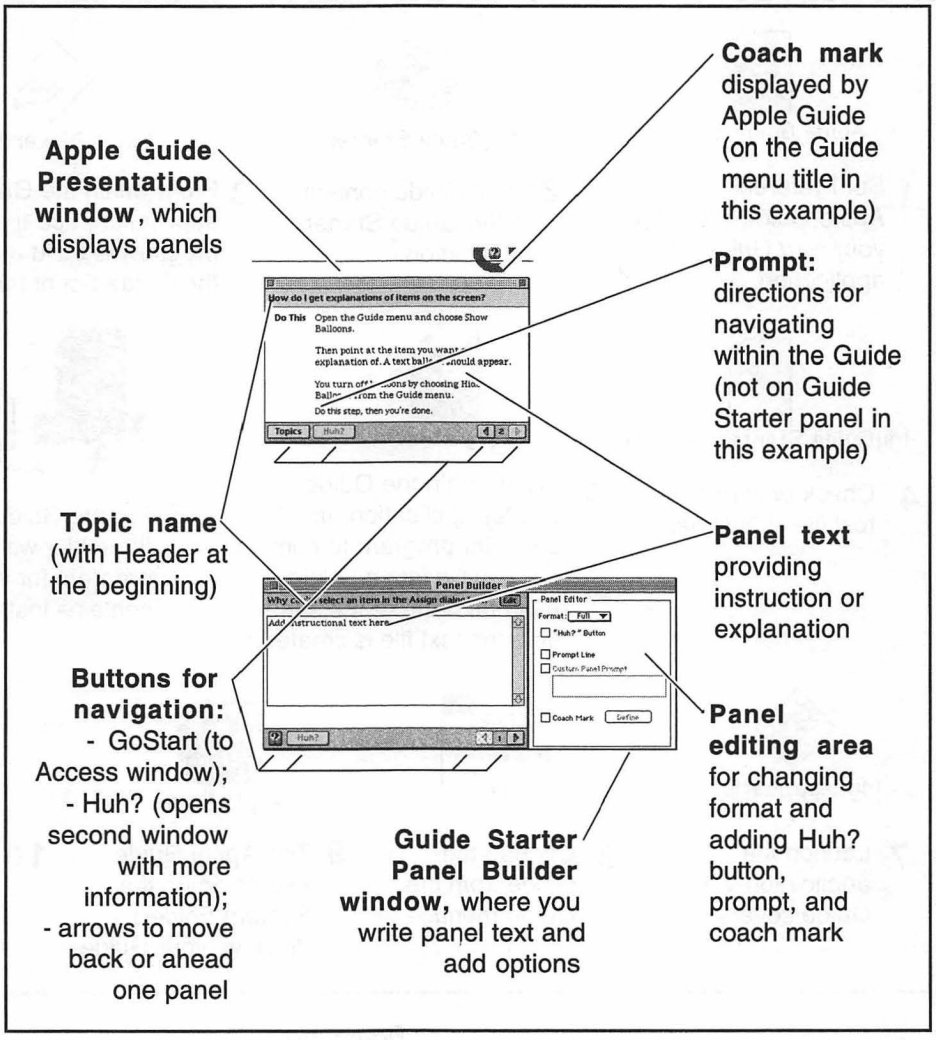


Figure 1-2
Common elements of Presentation window in a Guide and Guide Starter

associated with individual panels. After reading a panel and performing an action (if one is described), the user moves to another panel by clicking an arrow at the lower right-hand corner of the window. To return to the Access window, the user can click the Apple Guide icon (or the Topics button for Macintosh Guide, in Figure 1-2) at the lower left-hand corner.

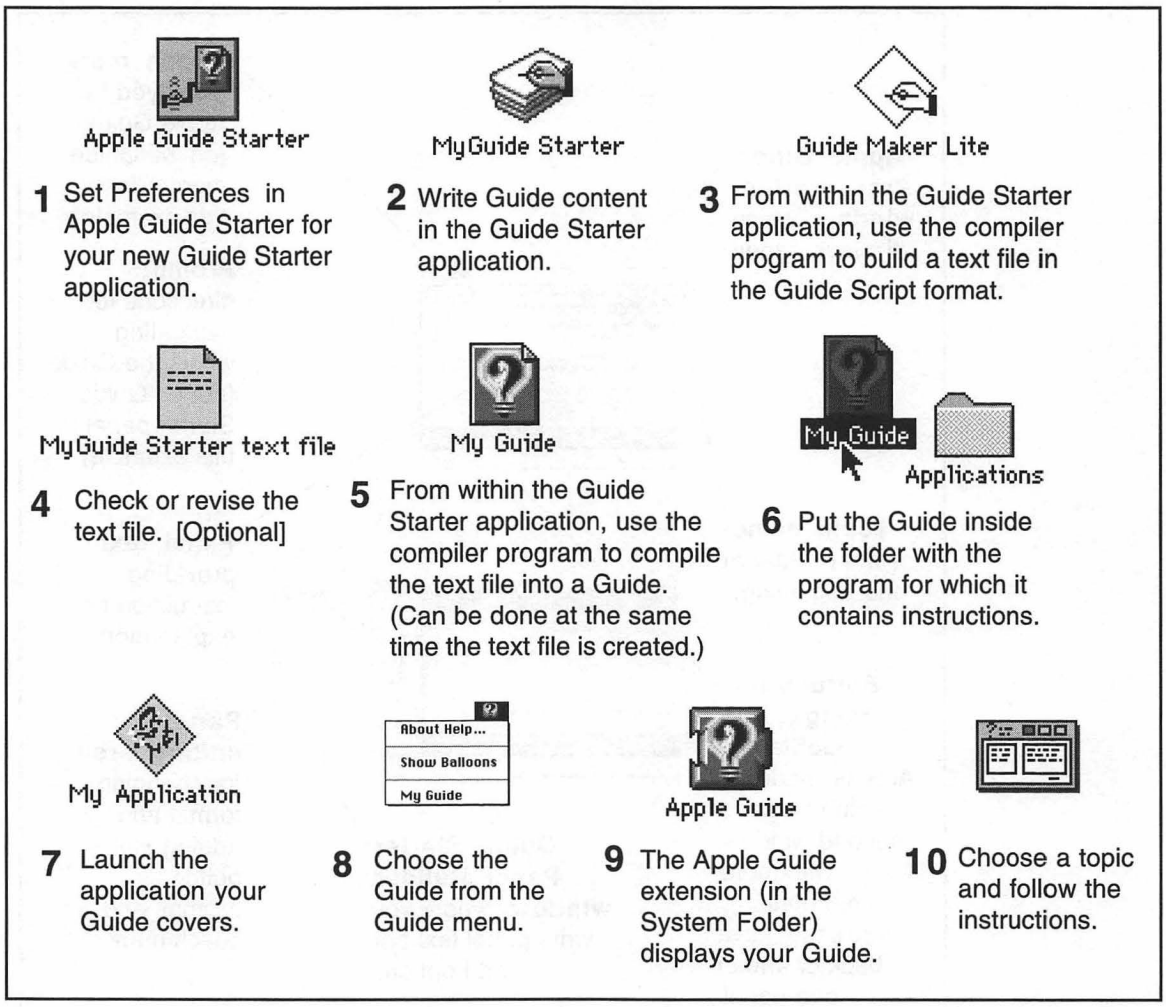


Figure 1-3
Major steps and components of the Guide-making process

Building a Guide

Figure 1-3 shows the components of the Apple Guide technology and the process you follow when you create a Guide database with Guide Starter.

Avoiding Confusion in Terminology

In this book, we've tried to keep the technical terms to a minimum. Still, you have to learn some new jargon whenever you acquire a set

of computer skills. And because you may go on from Apple Guide Starter to work with the software and documentation in the developer's package, *Apple Guide Complete* (Addison-Wesley, 1995), we've used essentially the same terminology employed in that technical book.

To help you sort out the similar terms, which can be confusing, here's a summary of the synonyms and special vocabulary in the Apple Guide lexicon:

- Apple Guide—the system software extension that displays and manipulates a Guide; also the general name for the on-screen instructional technology. (See Chapter 4 for a discussion of Apple Guide's key instructional features.)
- Apple Guide Starter—the program included with this book, which assists you with writing content for a Guide and then builds a text file and directs compiling of the Guide. For simplicity, we frequently refer to this program as Guide Starter. (See Chapter 2 for a hands-on introduction to Apple Guide Starter.)
- Guide—a collection of instructions and information that has been compiled to work with the Apple Guide extension; also used for the Guide menu (which is sometimes called the Help or Balloon Help menu). (See Chapter 2 for a hands-on introduction to Guide databases.)
- Database—in this book, the contents of a Guide. We use "Guide database" and "Guide file" interchangeably.
- Topic—one set of instructions, an explanation, or a definition; also called a task (when it involves doing something with the computer) or a sequence. (See Chapter 5 for a discussion of devising Topics for a Guide.)
- Sequence—the script, or computer language, that specifies what content is included in a topic (a set of instructions, an explanation, or a definition). (See Chapter 11 for an example of the sequences section in the text file for a Guide.)
- Guide Script—the name for the "markup language" or "tag language" that directs the compiler program as it creates a Guide database from the content you've written. (See Chapter 18 for an extended discussion of Guide Script.)
- Topic Area—a major subdivision of content in a Guide. (See Chapter 5 for a discussion of determining Topic Areas for a Guide.)

- Header—a phrase that's used as the prefix for a number of topics in a Guide; examples are "How do I" and "Definitions." (See Chapter 5 for a discussion of writing Headers for a Guide.)

To maintain consistency with the Apple Guide extension and the Guide Script language, the Guide Starter program uses most of the above terms in its interface and in the text file it generates.

What You Need to Create Guide Databases

You need a few basic items and skills to begin creating Guides. These include the following:

Hardware

- Macintosh (or compatible computer) with at least 8 megabytes (MB) of memory (RAM)

Software

- System 7.5 or a later version (with Apple Guide and AppleScript installed)
- Apple Guide Starter and Guide Maker Lite (from the Starter Kit Disk)
- SimpleText (distributed with System 7.5) or TeachText (distributed with earlier versions of Macintosh system software)
- A word processing program (optional, but necessary for working with text files larger than 32 kilobytes)

Skills

- Knowledge of the Macintosh (technical expertise or programming experience not required)
- Familiarity with the product or task you're creating a Guide for
- Experience or strong interest in writing step-by-step instructions

If you are not using System 7.5 (also named Mac OS 7.5 for some computers), you must install that version or a later one in order to work with Apple Guide and Guide Starter. You can purchase a System 7.5 upgrade kit, which is compatible with all Macintosh models, including the Macintosh Plus and PowerBook model 100.

What You'll Find in the Rest of This Book

We've designed the content of this book to provide quick access to all its instructions for creating Guide databases. In Chapters 4 through 10, which contain the primary instructions for planning and writing Guides with the Guide Starter program, the "hands-on" material appears at the beginning of the chapter (or at the beginning of a section in some instances) and is identified with a check mark (✓). The balance of the chapter is the "head work"—concepts, explanations, examples, and tips. Five of these chapters end with a "Guide Ideas" section, which describes specific types of Guides you can use for your business, school, or home.

Chapters 11 through 18 cover advanced features you can add to your Guides by revising the text file. These chapters focus on scripting and sophisticated Apple Guide technology, so the hands-on instructions are integrated with explanations of the features and syntax. Four of these chapters end with a "Guide Enhancements" section, which contains detailed instructions and script examples for implementing advanced features in sample Guide databases.

Here's a digest of the book's content:

Part I: Quick Start

Chapter 1
introduction to Apple Guide and the Apple Guide Starter Kit; basic terms; Macintosh system requirements

Chapter 2
brief hands-on orientation to Guide databases; short tutorial for Guide Starter

Chapter 3
condensed version of concepts and guidelines for creating on-screen instructions

Part II: Creating a Basic Guide with Guide Starter

Chapter 4
hands on—writing a plan for your Guide;

head work—principal considerations and analysis necessary to design a Guide database

Chapter 5
hands on—using Guide Starter; defining attributes for your Guide; entering Topic Areas, Headers, and Topics in the Access window

head work—matching the type of Guide to your content; determining the format for topics (questions or statements); fine-tuning the topics list

Chapter 6
hands on—planning panel content and doing the tasks; using the product your Guide covers

head work—answering a set of questions for each panel to determine its content; emphasizing basic Apple Guide features such as coach marks and closely related information or topics

Chapter 7
hands on—using Guide Starter; writing panels; adding panel options

head work—considerations for writing to meet the demands of online instruction; making efficient use of panels; anticipating users' needs for support information

Chapter 8
hands on—using Guide Starter; adding index terms in the Access window; assigning topics to each term

head work—determining keywords; becoming familiar with the Look For feature; fine-tuning index assignments

Chapter 9
hands on—using Guide Starter, building a text file; compiling a Guide database

head work—studying the text file to begin learning scripting

Chapter 10
hands on—using the new Guide; testing its content; making any necessary changes with Guide Starter

head work—observing users working with your Guide; analyzing all feedback to determine revisions

Part III: Adding Bells and Whistles

Chapter 11
using a word processor for efficient work with the text file; mechanics of copying, correcting, and archiving

Chapter 12
changing the type or style of coach marks used in a Guide

Chapter 13
adding scripting to display graphics and QuickTime movies in a Guide

Chapter 14
dividing topics into branches; adding scripting to create decision panels

Chapter 15
adding word lists to augment the Look For feature

Chapter 16
adding AppleScript components to the text file to augment coach marks and automate operations

Chapter 17
adding scripting to the text file to include standard Apple Guide context checks

Chapter 18
an introduction to the Guide Script language and its primary commands

Many of the examples and hands-on instructions in the book use sample Guides and text files on the Starter Kit disk.

You can get your own hands-on introduction to Apple Guide Starter and the Apple Guide technology by moving on to Chapter 2.

TWO

A HANDS-ON INTRODUCTION TO GUIDES AND GUIDE STARTER

The best preparations for creating your own Guide databases are to sample some of Apple Guide's key instructional features and to take a practice run with Guide Starter. This chapter gives you a chance to do both.

Getting Acquainted with Apple Guide

The Apple Guide technology built into the Macintosh system software (or Mac OS) consists of two parts: the Apple Guide extension, which is the "delivery engine" for on-screen help, and a group of compiled databases, or Guides, which provide the instructional content.

Types of Databases

A Guide can be one of the following database types:

- Help—the primary instructions for using a program or performing a set of tasks
- Tutorial—an introductory set of exercises, usually presented sequentially so that each task naturally leads to the next
- Other—a general database that can be used for any purpose

- Shortcuts—a summary or quick reference for common actions
- About—an explanation of the Guide currently in use (or of the items in the Guide menu)
- Mix-in—a database whose content is inserted into another Guide so that users see the combined information

With the exception of the Other database, the Guide menu can contain only one of each type of database. Other databases always appear at the bottom of the Guide menu, so many of these Guides can appear in the menu.

You can use Guide Starter to create any of these types of Guides except the Mix-in, which must be scripted specifically for inclusion in another database.

In most cases, a database is designed for use with a specific program. That database will appear in the Guide menu when its specified program is active, but it won't be in the menu when you switch to a program that isn't associated with it. For example, when you're using the Finder (the system software program that displays the desktop and manages your files), you'll see About Apple Guide, Macintosh Guide, and Shortcuts (and possibly others). When you open an application program, About Apple Guide and Macintosh Guide won't be visible, and the Guide or other help for the program will appear in the menu.

Guide Database Files in System 7.5

The system software (version 7.5 or later) for a Macintosh computer has a standard set of Guide databases. These include at least five of the six types of databases the Apple Guide extension supports, plus additional files for some Macintosh models. These five types of databases appear in the Guide menu (when PowerTalk, a separate component, is installed).

Figure 2-1 shows the Guide menu for a system with all possible components installed. (The Guide menu on your computer may not include all of these items. This is because the tutorial is not included as part of a software upgrade, and the PowerTalk Guide appears only after you perform the separate installation procedure required for the PowerTalk software.)

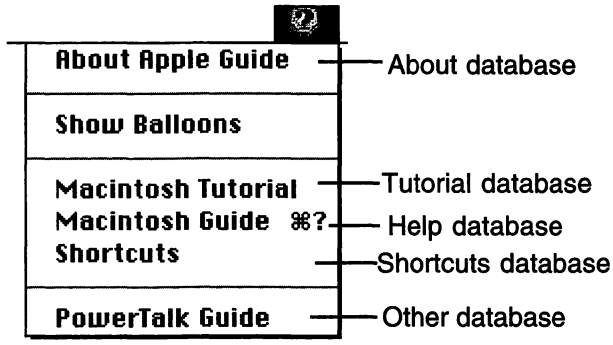


Figure 2-1
The Guide menu for a Macintosh with all components installed

The type of database not visible in the Guide menu is a Mix-in database. Its content is integrated into one of the other databases and appears to users as part of that database.

Table 2-1 shows the Guide files that are included with System 7.5 and installed on the appropriate Macintosh models. The table includes the type of each database, the number of each that can appear in the Guide menu at one time, and the application program that each database accompanies.

Table 2-1
Guides with System 7.5, including type, number possible, and target application

System 7.5 Guide Database	Database Type	Number	Used for Application
About Apple Guide	About	One	Finder
Apple Video Player Guide (certain models only)	Help	One	Apple Video Player
AppleMail Guide	Help	One	AppleMail
File Assistant Guide (PowerBooks)	Help	One	File Assistant
Macintosh Guide	Help	One	Finder
Macintosh Guide Additions* (for PowerBooks, AV video, speech)	Mix-in†	Multiple	Finder
Macintosh Tutorial	Tutorial	One	Finder
PowerTalk Guide	Other	Multiple	Finder
Shortcuts	Shortcuts	One	Finder

* Not visible in the Guide menu; its content is integrated into Macintosh Guide
† A complex database not covered in this book

Exploring Macintosh Guide

Macintosh Guide is a useful model for most Guides you're likely to create. Even if you've used this database already, take some time to explore the on-line instructions in Macintosh Guide. As you work with the database, try out its content by doing the following:

- Move ahead in the instructions without taking the action specified on a panel.
- Purposely take a wrong action.
- Get ahead of the instructions (by opening a folder or control panel before you reach the panel that prescribes that action).
- Take a control panel out of the System Folder and then select a task that uses the control panel.
- Check instructions for tasks you know how to do and other tasks you don't know how to do.
- Use all three methods of finding the information you want in the database—Topics, Index, and Look For.
- Look for some of the words or phrases you would naturally use to describe a task (as you look for information in the Access window).
- Take note of the terminology used in the Topics lists and Index.

By becoming familiar with the language and the structure of tasks in Macintosh Guide, you're likely to have an easier time when you begin writing instructions for your own Guides. If the features and style used in Macintosh Guide are useful for the tasks you want to document, take a similar approach. Chances are your users will benefit from the consistency between the Guides provided with the computer and the new ones you create.

To give you a closer look at the features and style of Macintosh Guide, the following are highlights from instructions for three of its tasks. Go through these instructions on the computer now.

Task 1: Topics/Reviewing the Basics/How Do I Use Macintosh Guide?

Follow these steps to sample the database:

1. Choose Macintosh Guide from the Guide menu.
2. Click the Topics button in the Access window.
3. Select "Reviewing the Basics" from the Topic Areas list on the left.

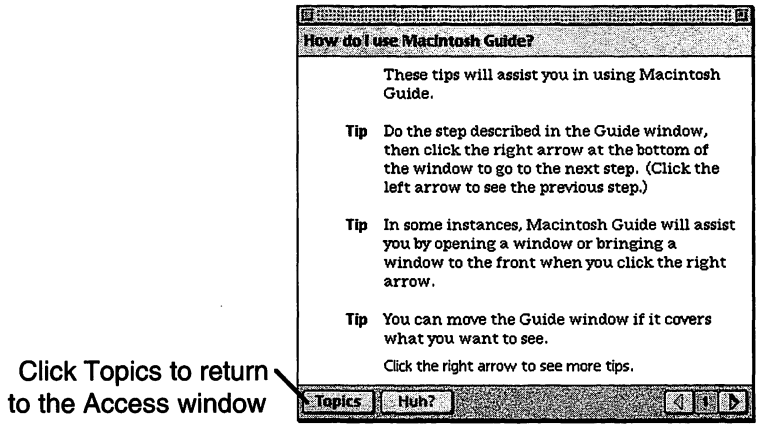


Figure 2-2
Presentation window from Macintosh Guide with Guide tips

4. Select "How do I use Macintosh Guide?" from the Topics list on the right and click OK.
5. Read the content of the first panel. Then click the right arrow to move ahead. (See Figure 2-2.)

   The dozen or so tips in this task provide useful information—for you as you plan a Guide and for users of your databases.
6. When you've read all five panels, click the Topics button (also called the Go Start button) to go back to the Access window.

Task 2: Index/"Diacritical Marks"/How Do I Change the Way Text Is Displayed?

1. Click the Index button in the Access window.
2. Click the letter "d" (or drag the slider to "d").
3. Select "diacritical marks" in the list of Index terms.
4. In the list of Topics on the right, select "How do I change the way text is displayed?" and click OK (Figure 2-3).
5. Read the introductory panel. Then move to the second panel.
6. Even though a coach mark shows you the menu to use, ignore the instruction on panel 2 and click the right arrow to move ahead.

   In a moment, you'll see another panel advising you that Apple Guide is taking the action prescribed by the previous instruction.

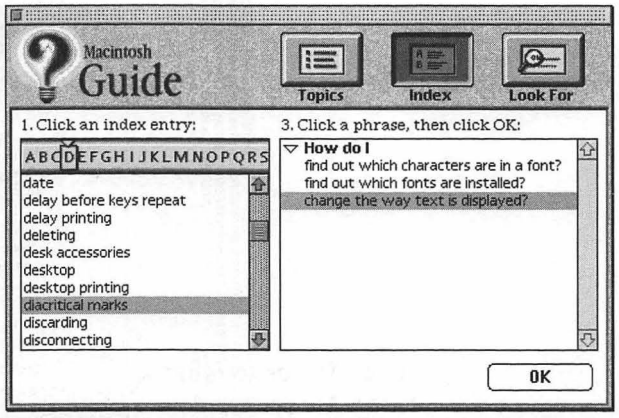


Figure 2-3
Index selections for "diacritical marks"

Figure 2-4 shows this panel. It remains on the screen until you click the Continue button (step 7) so that you have the optimal chance of noticing that something has changed on the screen.

7. Click Continue.

   The next panel appears, and in a few moments, the Text control panel is indicated with a coach mark (Figure 2-5).

   Should you again move ahead without opening the control panel, you'll see another Continue-button panel, and Apple Guide will open the control panel. This is the limit of the do-it-for-you actions in Macintosh Guide, however. Once the control panel is open, you need to decide which options you want to use or change.

8. Click Topics to return to the Access window.

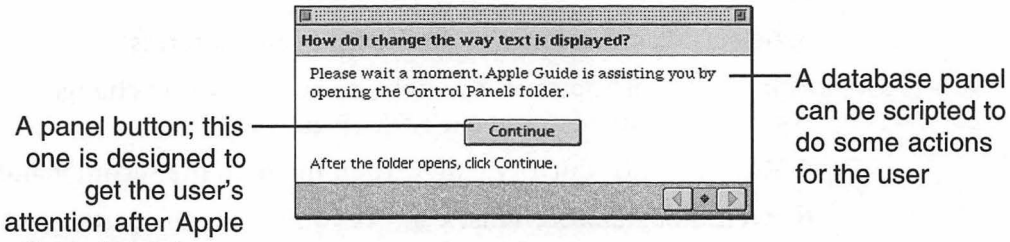


Figure 2-4
A panel alerting the user that Apple Guide is taking an action

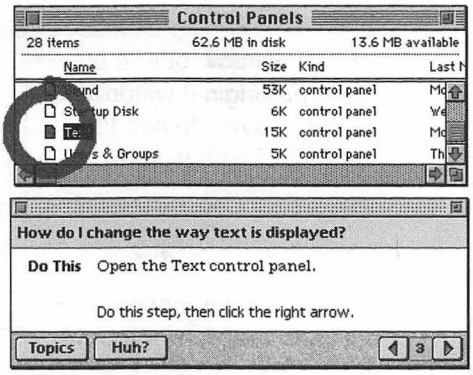


Figure 2-5
Apple Guide identifies a control panel with a coach mark

Task 3: Look For/Type "Keyboard"/How Do I Change the Keyboard Layout?

1. Click the Look For button in the Access window.
2. Click the arrow button, type *Keyboard*, and click Search.
3. In the list on the right, select "How do I change the keyboard layout?" and click OK.

   Notice that the first panel of this task contains two radio buttons (Figure 2-6).

   You can use radio buttons to build branches into a task, a technique that is essential if your Guide covers fairly complex software and you want to keep the total tasks in the database to a manageable number. The radio button that is selected when the panel appears should be the one most users are likely to choose.

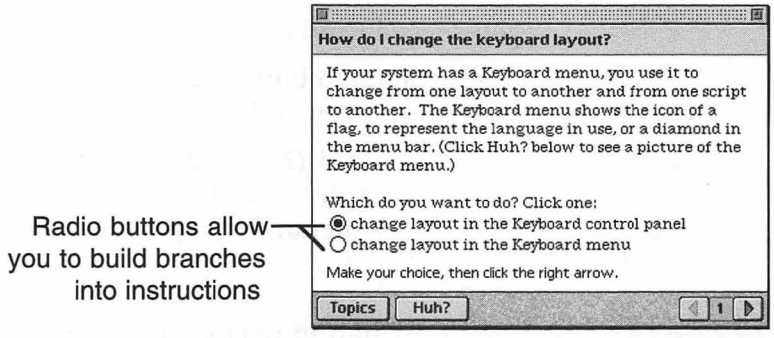


Figure 2-6
A panel with radio buttons that lead to separate branches of instructions

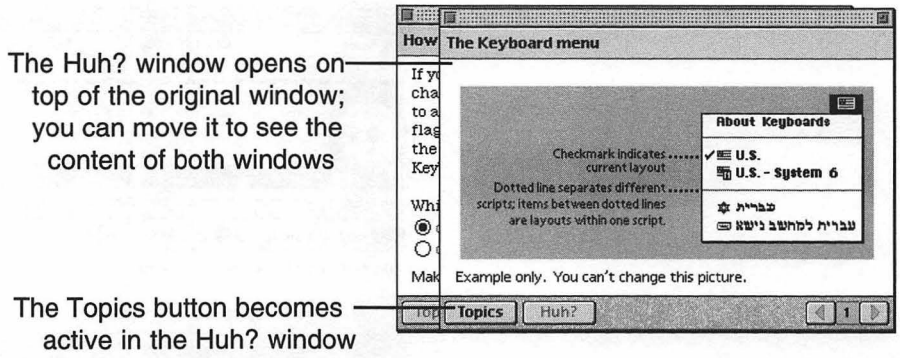


Figure 2-7
The Huh? window showing a picture for user's reference

Suppose you aren't certain whether you have a Keyboard menu or you're just curious to see that menu before choosing one of the radio buttons. You can use the Huh? button to open a second window that shows the Keyboard menu.

4. Click Huh? at the bottom of the panel.

   Figure 2-7 shows the Huh? window open in front of the original window.

   The Huh? button and its window are especially good for providing background information, such as a picture or a definition.

   You can also use the Huh? button to give users access to a task that's closely related to the current one. To see an example of this use of the Huh? button, continue with the following steps.

5. Click the close box of the top window (the one that opened when you clicked Huh?).

6. Click the radio button labeled "Change layout in the Keyboard menu" and then click the right arrow to move ahead.

   The next panel (Figure 2-8) describes the use of the Keyboard menu and notes that if the user doesn't see the desired layout listed in the menu, that layout must be installed.

7. Click Huh?

   The Huh? button opens the first panel of another task, again in front of the original window. To avoid confusing the two windows

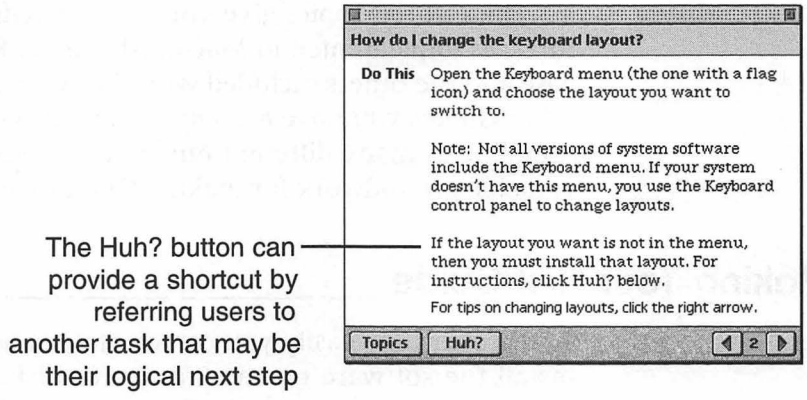


Figure 2-8
A panel that uses the Huh? button to lead to a closely related task

(although only the front one is active), you may want to compress the window of the original topic and move the Huh? window so that you can see the original Topic's name as you follow the steps in the related task.

8. Drag the Huh? window (on top) so that it isn't covering the original panel.
9. Click the zoom box (at the upper right-hand corner) of the original panel's window to compress it (Figure 2-9).
10. Click the close box of each window when you're finished.

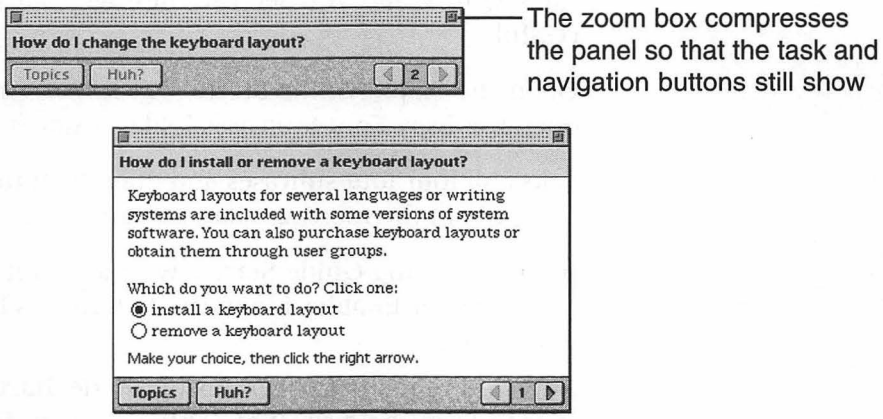


Figure 2-9
The Huh? window is active with the original Topic's window compressed

These three Topics give you some specifics of Apple Guide's features, as implemented in Macintosh Guide. Keep exploring this database and the others included with the system software, as well as any other Guides you have available. Using this online help system while sampling as many different implementations of it as possible lays invaluable groundwork for making Guides yourself.

Making Your First Guide

To discover how easily you can create a database with Guide Starter, install the software on the Starter Kit disk and use the program to write instructions for a task. Then compile them into a Guide.

Installing the Starter Kit Software

The software on the Starter Kit disk is compressed, but you can expand the files from the floppy disk onto your hard disk. Here's what to do:

1. Insert the Starter Kit disk in the floppy disk drive of your computer.
2. Double-click Apple Guide Starter.sea to open it and click Continue in the message that appears.
3. Choose a location on your hard disk for the expanded files and then click Save.
4. Click Quit when you see the message that installation was successful.
5. Open the Apple Guide Starter Kit folder on your hard disk and open the Espy Font Suitcases folder inside it.
6. Select all four font suitcases and drag them to your System Folder. Click OK in the message that appears.
7. If you are using Guide Starter with a Power Macintosh, drag the XTND Power Enabler file to the Extensions folder inside the System Folder.

Now you're ready to begin using Guide Starter and the tools provided with it on the disk. (See "Other Files in the Starter Kit Folder" later in this chapter for information about the software included with Guide Starter.)

Important: Do not move any other files out of the Apple Guide Starter Kit folder until you're thoroughly familiar with Guide Starter and the compiling process. Certain files (Guide Maker Lite, Standard Resources, and the logo art files) must be in the folder with Guide Starter for the program to compile a database. In addition, the Espy fonts must be in the Fonts folder in the System Folder.

Writing Instructions with Guide Starter

The process of creating a Guide with Apple Guide Starter involves the following four stages:

1. Set attributes for your own Guide Starter application.
2. Plan and write Guide content.
3. Prepare or edit a text file (using Guide Script tags).
4. Compile a text file into a Guide.

This hands-on exercise will take you through these four stages as you create a Guide that provides instructions for making a document into a template. (You'll also use a variation of this Guide for some examples of advanced Apple Guide features in Part III of this book.)

Configuring Your Own Guide Starter Application

When you begin a new Guide Starter project, you first specify several attributes for the Guide you're creating. The program acts as a template for this purpose. Once you complete the configuration in a series of Preferences screens, Guide Starter creates a duplicate application with the name and attributes you specify. You then use the new application to write your Guide content.

Follow these steps to configure the program for your Guide:

1. Open the Apple Guide Starter Kit folder (if necessary) and double-click the Apple Guide Starter icon to launch the program.

   A welcome screen appears, explaining that Guide Starter needs some information before it can create a Guide file.

2. Click Continue.

   A credit screen appears momentarily, followed by the first of four Preferences screens (Figure 2-10 shows the first one).

3. Review the information in Preferences 1. Don't change the settings. Then click Next.

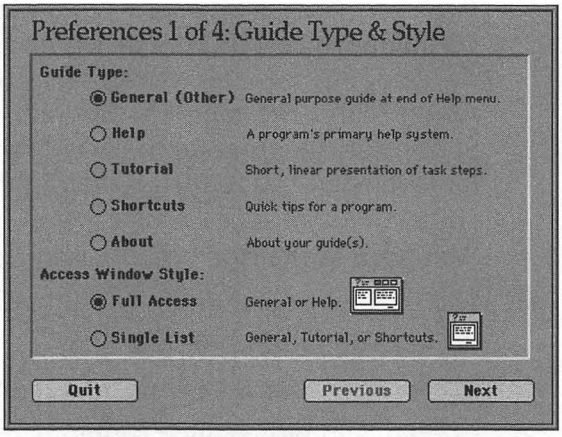


Figure 2-10
One of Guide Starter's Preferences screens

4. In Preferences 2, add a title for your Guide by selecting "«Task»" and typing *Template-making* before the word "Guide". Click Next.
5. In Preferences 3, select the words "personal help" and type *Template-making Guide*. Click Next.
6. In Preferences 4, click Select Folder for Database and use the directory dialog box to locate the Apple Guide Starter Kit folder. When the folder is showing in the pop-up menu (meaning it is open), click Choose Apple Guide Starter Kit folder at the bottom of the dialog box. (The folder name will be shortened to fit.) Click Done.

   After the Preferences screen clears, the directory dialog box appears.
7. Select the word "My" in the text box. Type *Template* and add a space so that the name for the new Guide Starter application is "Template Guide Starter." Click Save.

   After a few seconds, the Access Window Builder of the new application appears (Figure 2-11).

This program is a duplicate of Guide Starter, except for the changes you made in the Preferences. You use the new application to design and write the content for a specific Guide file, to build a Guide Script text file, and to compile a finished Guide. You can always open this

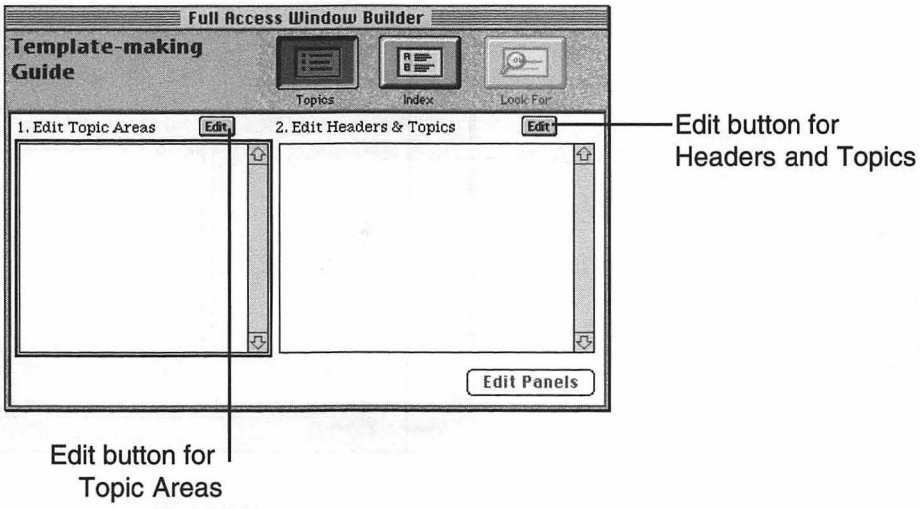


Figure 2-11
The Access Window Builder showing a double-list format

application, change the content, and then recompile to get a revised Guide.

Entering Topic Area, Header, and Topic

Now you're ready to enter content for your new Guide. First, you need to put the relevant information in the Access window (see Figure 2-11).

Don't worry about saving your work. Guide Starter saves it automatically.

1. Click the Edit button above the list box on the left (Topic Areas).

   An extension appears at the bottom of the window.

2. Click in the text box that appears at the bottom of the window and then type *Files*. Click Add.

3. Click Files in the Topic Area list to select it. Then click the Edit button above the list box on the right (Headers & Topics).

   The extension appears at the bottom of the window with the Header button selected.

4. Click in the text box at the bottom of the window and type *How do I* for the task. Click Add.

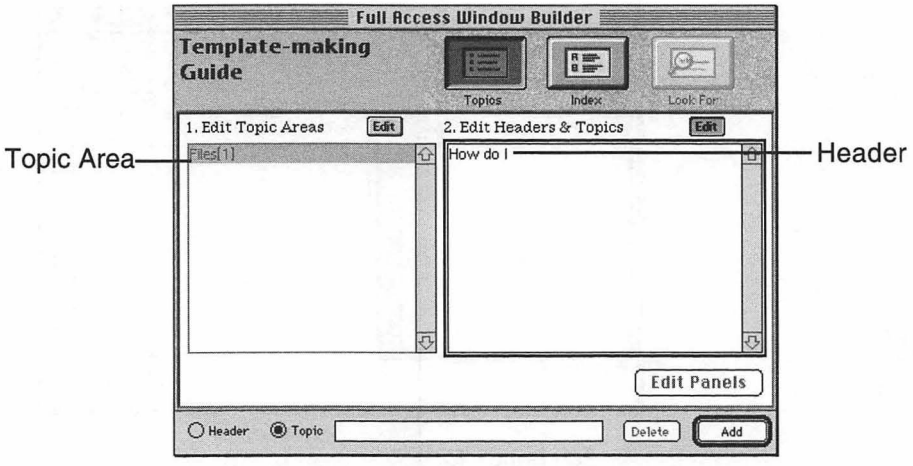


Figure 2-12
The Topic Area and Header in the Guide Starter application

The window should now look like the one in Figure 2-12.

5. Make sure the Topic radio button is selected and then type *create a template?* Click Add.
6. Click the Edit button at the top right to deselect it. Then select the Topic "Create a template?" on the right. (Also check to see that Files is selected on the left. If not, select it.)

   When both the Topic Area and the Topic are selected, a dialog box appears, asking if you want to assign an existing panel or create a new panel.

7. Click New.

   A new window—Panel Builder—appears on top of the Access Window Builder (Figure 2-13).

Entering Steps for the Task

Now determine what the actual steps—the panels—will say in the Panel Builder window (see Figure 2-13).

1. Click the Edit button at the upper right-hand corner to display the panel options (Figure 2-14).

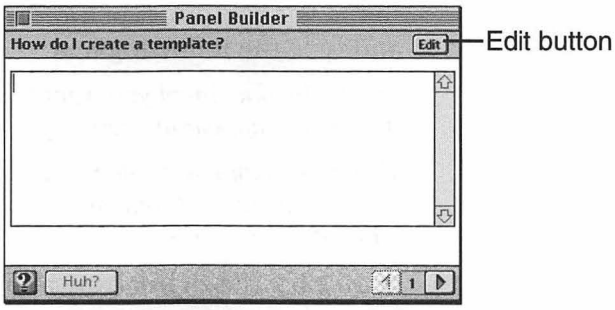


Figure 2-13
The Panel Builder window for new content

2. Type the following text for the first panel of instructions:

   You can create a template in either of two ways. Many programs provide a template option (called Stationery) when you save a document. Or you can use the Info window for a document to designate it as a Stationery pad.

   Then click the Prompt Line box (in the Panel Editor section at the right) to put an X in it.

   Guide Starter's standard prompt appears at the bottom of the window.

3. Click the right arrow (at the lower right) and then click New in the dialog box.
4. In the Format pop-up menu (in the Panel Editor section at the right), choose Tag.

   The text area changes to a narrow column on the left and a wider one on the right.

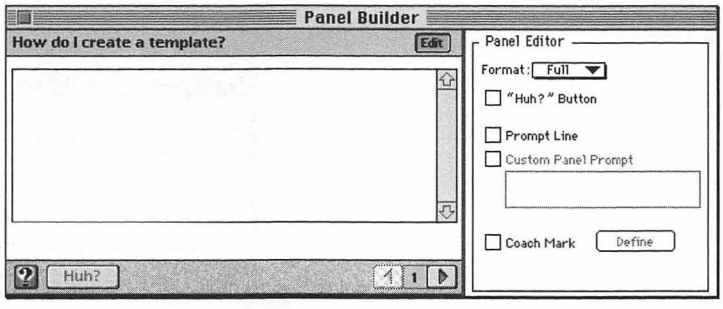


Figure 2-14
The extended Panel Builder window, with options section showing

5. In the left column, type *Do This* and then click in the right column and type the following text:

   Create the document you want to use as a template. Include all content that should appear in every copy.

   When the template content is complete, save the document with a name such as "Template Original." (This is the version you can use to revise the template later.)

   Then click the right arrow and click New in the dialog box.

6. Type the following text in the right column of the new panel:

   Now save a second copy of the document by choosing Save As from the File menu.

7. In the Panel Editor, click the Coach Mark box to put an X in it. Then click Define.

   The Coach Mark Builder window appears (Figure 2-15).

8. Choose Menu from the Coach Mark Style pop-up menu.

   The window changes to show two text boxes.

9. Type *File* in the Menu Name box and *Save As...* in the Menu Item Name box (see Figure 2-16).

   Note: You must use the key combination Option-semicolon for the ellipsis in a menu item, like "Save As" above. If you type three periods instead of Option-semicolon, the coach mark won't appear on the item (because the text you enter in Coach Mark Builder must match the menu item exactly).

10. Click OK after you type the menu name and menu item text. Then click the right arrow and click New in the dialog box.

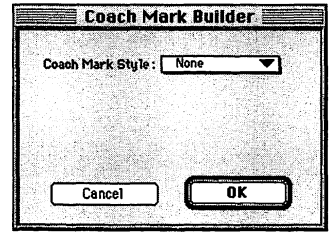


Figure 2-15
The Coach Mark Builder window

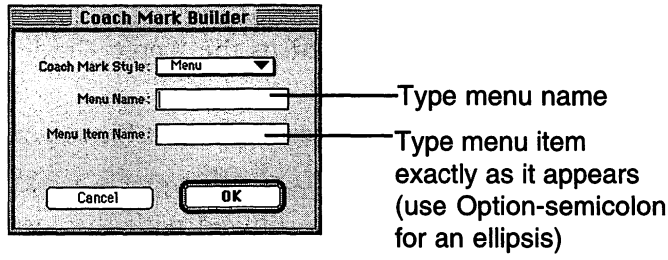


Figure 2-16
The window for specifying a coach mark on a menu

11. Type the following text in the new panel:

    Check the Save dialog box to see if there's a Stationery option (usually in a Format pop-up menu). Choose Stationery if you find it.

    Change the document's name and click Save.

    Then click the right arrow and click New in the dialog box.

12. Type the following text in the new panel:

    If you did not use a Stationery format when you saved, locate the document's icon (in the Finder) and click to select the icon.

    Then click the right arrow and click New in the dialog box.

13. Type the following text in the new panel:

    Choose Get Info from the File menu.

    Then click the Coach Mark box to put an X in it and add a coach mark on Get Info in the File menu.

    Review steps 7, 8, and 9 if you need help with the coach mark.

14. Click the right arrow and then click New in the dialog box.

15. Type the following text in the new panel:

    In the Info window, click to place an X in the box labeled "Stationery pad" at the lower right-hand corner of the window.

Building a Text File

When your content is complete, you can have Guide Starter create a text file as follows:

1. Choose Build File from the File menu.

   The directory dialog box appears.

2. Change the name in the text box if you wish. Make sure the file will be saved in the Apple Guide Starter Kit folder. Then click Save.

   In a few seconds the text file appears in the folder.

You can open the text file and see how the Guide Script tags and formatting are arranged with the instructions you wrote. Studying files of scripts such as this is the best way to become familiar with the specialized syntax required to create a Guide from a text file. As you become more experienced at writing Guide content and using Apple Guide's features, you'll be able to copy or adapt portions of existing text files for new purposes.

Compiling a Guide

The final stage is compiling a Guide from the text file. (You can actually go directly to the compiling stage when the content is complete if you don't want to look over the text file before compiling.)

1. Choose Build Database from the File menu.
2. Type a name for the text file (if you want to change Guide Starter's suggested name). Click Save after you've specified the name and location you want. (You can click Replace if you see a dialog box asking whether you want to replace a file with the same name.)

   After a few moments you'll see a message reporting the progress of compiling followed by a message telling you that compilation was successful. Click OK.

Checking the New Guide

You can check the instructions in your Guide as follows:

1. Open the Apple Guide Starter Kit folder and locate your new Guide.

   A log file should be with the Guide—it is a SimpleText file with the same name as the Guide, plus ".err" at the end. The log file provides a report about the compile procedure.

2. Locate the SimpleText program and move or copy it to the Apple Guide Starter Kit folder.

   You'll use SimpleText to check the instructions in your new Guide. A Guide for an application program must be in the same folder

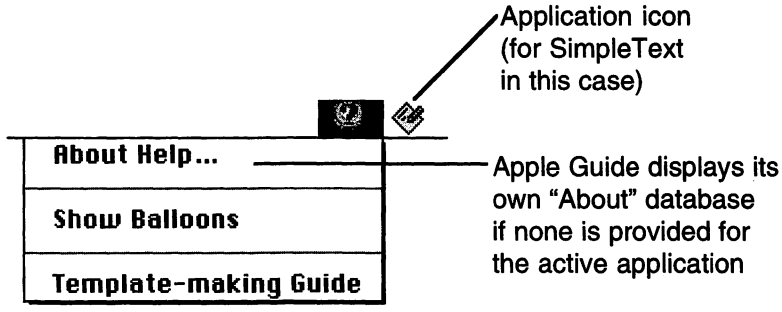


Figure 2-17
The Template-making Guide is available when SimpleText is active

with the program its instructions cover. Because the Template-making Guide and SimpleText are in the same folder, the Guide will be available (in the Guide menu) whenever SimpleText is the active program.

3. Open the SimpleText program and type some text in the untitled document.
4. Open the Guide menu and choose Template-making Guide (Figure 2-17).
5. Go through the instructions for creating a template, using the open document in SimpleText.
6. When you finish checking the Guide's steps, quit SimpleText and click the desktop to make sure the Finder is active. Then open the Guide menu.

   When the Finder is active, the system's Guides appear in the menu and the Template-making Guide doesn't appear (Figure 2-18).

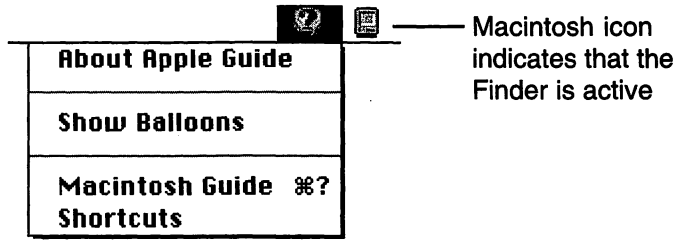


Figure 2-18
The Guide menu with standard system Guides

Other Files in the Starter Kit Folder

The Apple Guide Starter Kit folder you recently installed contains a number of files that will help you build databases and learn scripting. In addition to Apple Guide Starter, the folder has the following:

- Guide Maker Lite, the compiler program that Guide Starter uses to compile a Guide
- File Dynamo, a versatile program interface with which you can practice applying Apple Guide features
- a Guide with complete instructions for using Guide Starter, along with the text file for that database
- sample Guide files that demonstrate different types of Guides and their features
- sample text files that show examples of Guide Script syntax, including specialized scripting modules you can copy or adapt to add advanced features to your Guides
- graphics files, in PICT format, that you can use as templates to create custom logo art for your Guides
- the set of Espy fonts, which are specially designed for readability on the computer screen
- a Standard Resources file, which contains some items the compiler program needs, such as art for the buttons that are part of a Guide's interface
- XTND Power Enabler, an Extensions folder library file for using Guide Maker Lite on a Power Macintosh

File Dynamo

File Dynamo is the user interface portion of a program that generates little applications for handling files in a variety of ways. This program is also excellent as a practice vehicle for designing Guide files. The File Dynamo interface offers a few dozen options, thereby making it ideal for the targeted instructions and coach marks available with Apple Guide. Follow these steps to become familiar with File Dynamo:

1. Locate File Dynamo inside the Apple Guide Starter Kit folder and open it.
2. Click the tiny flag to the right of Set Attributes to expand the File Dynamo window (Figure 2-19).

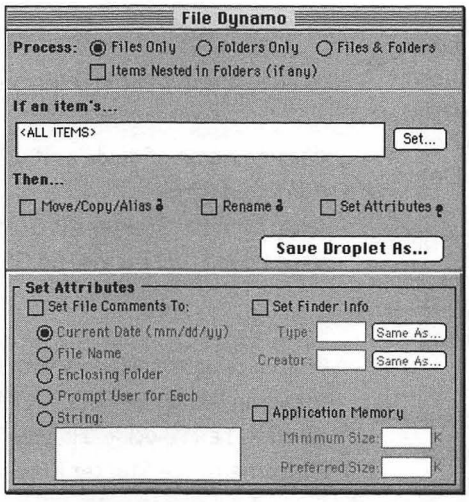


Figure 2-19
The expanded File Dynamo window with one set of interface options

Notice the several items that contain ellipses (...). When a user clicks one of these areas, such as Set... or Same As..., a dialog box appears on top of the program window.

In fact, the Same As... buttons in Figure 2-19 open the directory dialog box (also called "standard file"), a disk-and-folder-navigation aid that perplexes many inexperienced Macintosh users. You could get lots of practice with Apple Guide instructions—and do a big favor for lots of Mac novices—by writing a Guide that steers users through this maze of buttons, lists, and pop-up menus.

3. Browse File Dynamo as much as you wish. Quit the program by choosing Quit from the File menu.

A Guide's Text File

In addition to the Starter Kit Guide, the folder contains the text file from which the Starter Kit Guide was compiled. You can print this file and use it as a model for augmenting your Guide designs, or you can change or add content to it as you develop Guide Starter expertise.

Figure 2-20 shows a small portion of the Starter Kit Guide text file.

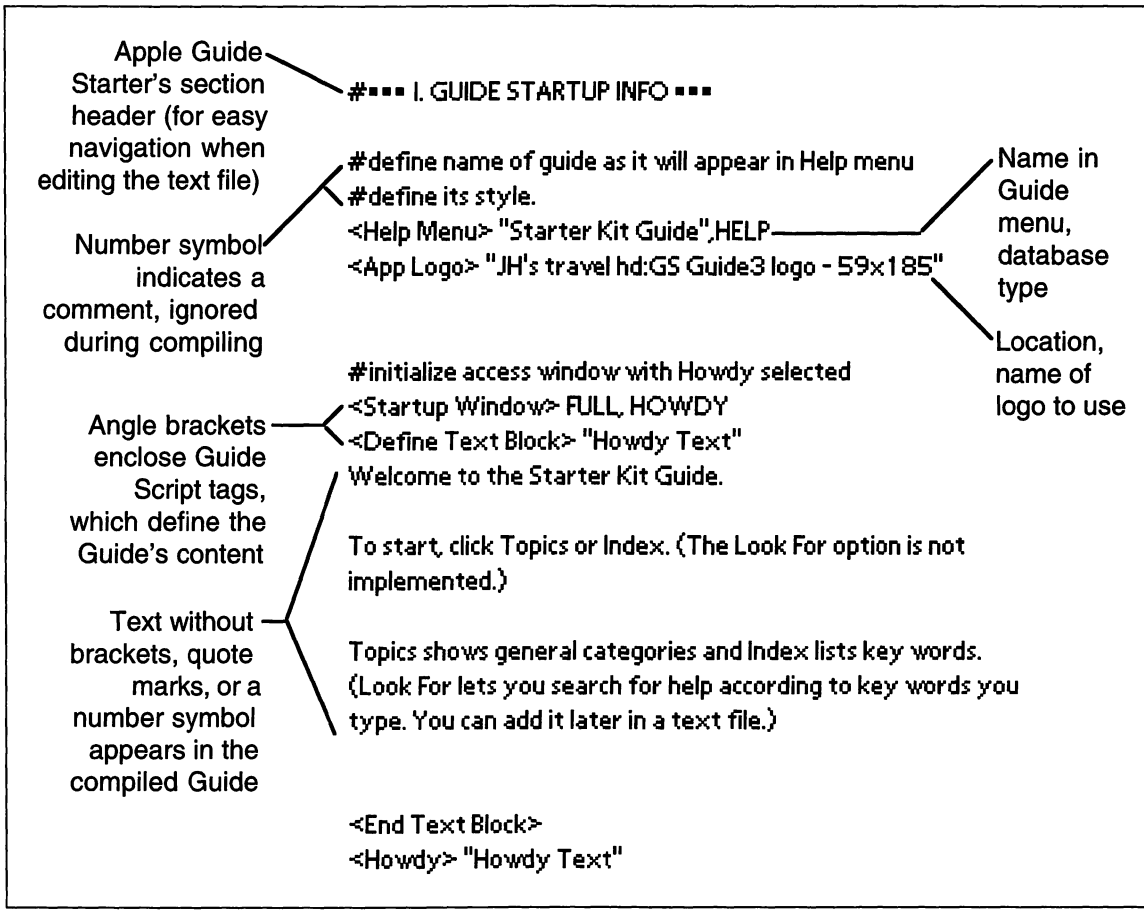


Figure 2-20
The first part of the Guide Script text file for a Guide

Chapters 11 through 18 in Part III, "Adding Bells and Whistles," use several text files to present such advanced Apple Guide features as using graphics or QuickTime movies, creating scripts with branches and decision panels, and building intelligence into Guides by adding context checks and AppleScript scripts.

How the Guide Starter Kit Builds Guide Files

As noted, the Guide Starter program generates text files with Guide Script tags in them and uses the compile component of Guide Maker to compile a Guide. Guide Starter also

- simplifies your work by automating much of the Guide-authoring process,

- streamlines Guide preparation by concentrating on the Apple Guide features likely to be used most frequently, and
- allows you to modify the text files it generates to add Apple Guide features that are less commonly used or more complex.

Guide Starter's Automatic Features

For a simple Guide file, Guide Starter automates everything but the writing of instructional content. (You're responsible for the content.) The features the program can build into a Guide automatically include the following:

- Choice of five types of Guide files—Help, Other, Tutorial, Shortcuts, and About
- Choice of double-list (Full) or Single-list Access window listing
- Option to use custom art for the logo in the Access window
- Entry of content for Access and Presentation windows in a WYSIWYG format
- Topics and Index information access in the Access window
- Standard panel formats (Presentation window), including Huh? button, Go Start (Topics) button, and prompts (custom, standard, or none)
- Standard coach marks (red circle, or red underline in menu) on menus and windows. Also automated designation of corners for a coach mark inside a window. (You click at three places, and the program calculates the coach's location.)
- Assignment of existing panels or sequences of panels while you are writing content (from a list showing all sequences and the first line of all panel text)
- Insertion of a new panel or deletion of an existing panel
- Generation of a Guide Script text file, including all instructional content entered and all features specified
- Compilation of a text file into a completed Guide

Chapters 4 through 10 provide detailed explanations and instructions for using Apple Guide Starter to create Guide databases with these features.

When using Apple Guide Starter, you enter instructional content and set options in a copy of the program (a separate application that

you name). You can at any time open this application and revise your content and the options you've set. (You can also open the text file that's generated by the program and add or change content and options in it.)

Features You Add in a Text File

To keep the process of using Guide Starter simple and fast, some of the features available with Apple Guide are not included in the program's built-in options. You can add these features by making simple changes in the text file and then using Guide Starter to recompile your modified text file into a Guide. The features you need to add by modifying the text file include the following:

- The Look For access method (with lists of terms it recognizes or skips)
- Variations in the style of coach marks (X or arrow)
- Graphics or QuickTime movies as part of Guide content
- Radio buttons or checkboxes to build branches into a task
- AppleScript modules to locate items to open them or indicate them with a coach mark
- Context checks to determine the computer's status (such as whether a specific window is open or active)

Chapters 11 through 17 provide details on how to add these advanced Apple Guide features in a text file. Chapter 18 offers an introduction to the Guide Script language.

Move ahead to Chapter 3 for a quick look at the thinking process involved in designing and writing Guide databases.

THREE
GUIDE-THINKING: A SHORT COURSE

Now that you've used Guide Starter to create a one-task database, you should be familiar with the mechanics of Guide making. That mechanical procedure has a corresponding thought process—the head work involved in planning, writing, and revising a Guide.

The tips in this chapter offer a condensed version of the Guide-thinking process. If you're eager to start building your own Guides, review these tips and then fire up Guide Starter. For more detailed explanations of the suggestions presented here, read the chapters that are referenced with each tip.

The Thought Process for Creating a Guide

At its most succinct, the head work for creating a Guide goes like this: Know. Be. Do. Redo.

A less cryptic way to describe Guide-thinking is that it consists of the following four stages:

1. planning (tips 1 to 4)—the *know* stage, in which you research the primary factors that influence the instructions you're developing
2. writing (tips 5 to 8)—the *be* stage, in which you craft those instructions using the knowledge you've gained

3. scripting (tip 9)—the *do* stage, in which you build intelligence into your instructions
4. testing and revising (tip 10)—the *redo* stage, in which you watch, listen, and refine

Ten Tips for Good Guides

These ten tips offer a good review of the major considerations in fashioning on-line instruction. As you work with Guide Starter, return to them whenever you want a quick refresher on how to use the computer to explain the computer.

Tip 1: Know Apple Guide's Strengths

Apple Guide is designed specifically for the following characteristics of people's behavior (see Chapter 4):

- We learn by doing—the physical connection to memory.
- We want to do our own work, and most of us are not curious about or interested in the computer.
- We want online help that's easy to use; we don't want to go back and forth from help screen to a program, and we don't want to try to memorize steps before the window closes.
- We ask for help only when we're stumped; otherwise, don't bother us.
- We have wildly different notions of what we want to do with the computer and highly individual vocabularies for describing what we're doing and what went wrong.

Tip 2: Know the Demands of Online Instruction

Extensive research and user testing have shown that special factors affect people when they use online help (see Chapter 4), as follows:

- After a short while of using a computer or a specific program, users develop a set of expectations about how the computer or program will work, where to look for menu commands, and so on.
- Users don't read much on the screen and generally are impatient to keep moving ahead.
- While you can leaf through a book and see how much it holds, you can't do that with an interactive computer help file. So the online help should give plenty of clues to what it contains.

- Certain information can't be delivered on-screen exclusively; for example, set-up, troubleshooting, hard disk formatting, and turning on the computer.

Tip 3: Know Your Audience

Every Guide database has a definable audience whose needs and experience should be central to its instructional design (see Chapter 4). For that purpose, you need to

- find out users' experience and confidence level,
- ask how much of their work is involved with the product that the Guide covers, and
- determine the type of equipment, networks, and software the Guide's users work with so that you can ensure the database will work on and support all systems.

Tip 4: Know Your Product's Compatibility with Apple Guide

Gather some technical information about the product or task your Guide covers so that you can take advantage of Apple Guide's features. You need to know the following:

- Do coach marks work with the software (see Chapter 4)?
- What sorts of context checks would be useful for the product, and can Apple Guide get the relevant information (see Chapter 17)?
- Is the software for the product scriptable (see Chapter 16)?

If one or more of the above conditions cannot be met, you'll need to devise ways around those constraints (see Chapter 6).

Tip 5: Be Brief

Here's where you meet the challenge presented by people's learning characteristics and their quirks when using online help (see Chapter 7). You'll want to do the following:

- Write specific, short steps and explanations.
- Include only one step per panel.
- Provide only the information necessary for each step. If background details are necessary for some users, add them by means

of the Huh? button (so that the users who need them can open the Huh? window, but others won't have to wade through the extra information).

Tip 6: Be Consistent

Live up to those users' expectations by doing the following (see Chapters 6 and 7):

- Adopt a standard structure for task instructions and other common features of your Guide.
- Use prompts consistently for navigation through the Guide.
- Keep the language the same for similar activities. For example, don't say "Select the Print command from the File menu" in one task and "Open the File menu and choose Print" in another task.

Tip 7: Tell Users What's Coming

Users see only a fraction of the help's contents at one time, so build in maps and connections like the following (see Chapters 6 and 7):

- Summarize a task on the first panel and include the principal action so that confident users can get the information they need from one panel.
- If a feature's usefulness or function isn't obvious—such as an alias—tell users what they can do with it and how it will help them.
- Use cross-references and tips whenever possible. If the product your Guide covers is a spreadsheet, for example, list the related tasks dealing with functions or formula-building on a panel attached to the Huh? button or at the end of a task.

Tip 8: Don't Assume Everything Is OK

With most help systems (and all manuals), you can't know what the user just did on the computer. With Apple Guide, you can, so put this intelligence to good use. Keep the following in mind:

- Apple Guide lets you guide users with coach marks and rescue them (or keep them on track) with context checks. Use these aids whenever feasible. Both are mandatory for tutorials or task-based Guides for novice users (see Chapters 6 and 17).

- If you don't have context checks available, be sure to think of the one, two, or three things that are most likely to go wrong or to confuse a user for the task (or for each panel) and try to build in the information they'll need to get past the problem. Use the Huh? button for general information, such as "If the window disappeared, try..." (see Chapter 6).
- If your Guide contains tasks, add a troubleshooting section. Many tasks have a logical "Why can't I" explanation that you can provide (see Chapter 6).

Tip 9: Syntax—It's the Mosquitoes

Scripting requires precision, which can be elusive. Keep in mind the following:

- There's an old saying about the jungle: "It's not the elephants that'll get you; it's the mosquitoes." The bugs that may get you after working in the Guide Starter text file include misplaced punctuation or quotation marks, cutting and pasting imperfectly, leaving out a crucial line or word. Take care! (See Chapter 11.)
- Before compiling, go through all tasks yourself. Make sure (1) content is accurate, and (2) spelling and typing are correct. Then check the text file for errors. (The more often you do it, the more familiar scripting will become.) (See Chapters 9 and 11.)
- You can create "compare" files or search routines to locate lost punctuation or quotation marks or missing angle brackets that will knock off a compile. Run those before compiling (see Chapter 11).

Tip 10: Respond to Users' Feedback

When there's still time to revise, gather feedback and use it (see Chapter 10). Do the following:

- If possible, get actual users or people with similar experience to try out your Guide. Observe them and have them tell you their questions and points of confusion. But don't give them answers. Instead, encourage them to use the Guide for the information they need, and to tell you what they're thinking as they work. Watching people use your Guide is humbling, but the feedback you get is invaluable.

- Take this information seriously, especially if three or more people have trouble with the same task or category in the Guide. Revise to improve navigation, clarity, and ease of use.
- Use the Guide yourself with the product. Test all instructions after each major revision of the Guide or the product.

And One More

This isn't so much a tip as a plea: Don't leave the Index or the assignment of tasks and Topics to Index terms and Topic Areas until the end of your Guide-building process. Remember that a user's access to your instructions is only as good as the entry points you provide. (See Chapter 8 for complete information about the Index.)

Try to establish a regular interval at which you review all Topics listed in the Access window and all Index entries and augment the assignments and terms whenever you add or change content. If you assign tasks and Topics only at the beginning and end of Guide development, you're sure to miss some useful—perhaps even obvious—connections.

Enough said.

PART II

CREATING A BASIC GUIDE WITH GUIDE STARTER

FOUR
DESIGNING YOUR GUIDE

Once you're familiar with Apple Guide and the contents of the Starter Kit disk, you can begin designing your own Guide databases.

Hands On: Getting Ready to Write

This chapter focuses on designing a Guide, so it contains more research and planning than actual work with the computer. Still, a written plan is essential, and there's a lot you need to find out. Your preparations include the following:

- Researching the audience for the Guide and the computing environment in which they work
- Using the product that your Guide will cover to analyze the tasks for which you need to write instructions
- Listing the Topics and major categories of information for the database

Head Work: Preparing for Guide Decisions

As you begin to design your Guide, it's natural to concentrate on the subject matter for the Guide and the mechanics of using the Guide Starter

program. But first take a few moments to step back—mentally, at least—and consider the big picture. You'll have plenty of time later to dive into the details.

By first doing some research and planning, you can make your authoring work easier and your Guide files more effective than they might be if you simply plunged in and started writing content. It's important to realize that creating a Guide involves a series of decisions. When you make the first one—choosing the format of the Access window, for example—you set the course for choices, features, and limitations in that database.

Designing and writing a Guide is essentially a linear process, so up-front investigation and thinking are crucial. The more thoroughly you can plot out your project and its content in advance, the greater are your chances of creating a useful product with a minimum of surprises or snafus along the way.

Here's what we suggest you look for in the big picture:

- Good models—existing Guides or other types of online help that provide excellent instructions
- Good matches—effective ways to use Apple Guide's key features, such as coach marks and interactivity, while meeting the user's general needs from online instruction
- Good Guide definition—a clear understanding of your Guide's purpose
- Good fit—instructions and knowledge level that are appropriate to your Guide's audience and the environment in which the Guide will be used
- Good Guide—this is where the details come in

You can make the most informed decisions about your Guide's design and content if you are thoroughly familiar with the features of Apple Guide. Similarly, becoming familiar with creative implementations of these features will help you write versatile and usable instructions.

Continue Studying and Using Guides

A number of software vendors and shareware creators have released Guide databases with their products. At least one major word processing program, Novell's WordPerfect 3.1, includes a Guide. Independent developers have also prepared databases: for Internet software,

such as one for Mosaic from the Macintosh community at the University of Michigan, and the excellent Guide (by Quinn "The Eskimo," the author claims) for Peter N. Lewis's Anarchie. Guides are also included with DeltaPoint's DeltaGraph Pro 3.5 program and WorldWrite, a new word processor from World Software Corp. that supports multiple language scripts, such as Arabic and kanji. Apple's PhotoFlash 2.0 also has a Guide.

Databases such as these, as well as those included with the Macintosh, can be useful models for the instructions and information you provide online. Staying current with this growing body of electronic documentation and sampling new Guides as they appear will introduce you to the innovations that authors are certain to bring to this young technology.

Understand the Strengths of Apple Guide

The Apple Guide project was initiated by the Instructional Products group at Apple, the folks who write user documentation. The design and features of this technology have their roots in extensive research related to computer users' needs for information and their styles of learning. The Apple Guide interface and content were repeatedly tested by users, revised, and retested over the course of four years.

The Apple Guide design responds to a set of fundamental problems and learning characteristics and incorporates a variety of solutions. The basic design assumptions are noted below, along with the instructional responses Apple Guide offers.

People Learn by Doing

Because Apple Guide is interactive, users can take an action while reading the instructions, without interrupting or losing their work.

People Want to Do Their Own Work with the Computer

Most users aren't interested in studying the computer itself; consequently, they are likely to reach an impasse sometime, at which point they'll need instructions.

People Need Online Help That Is Simple and Seamless

Straightforward navigation and persistence at the top layer make the Apple Guide window easy to use, and coach marks integrate help with the interface.

People Generally Ask a Question When They Encounter Difficulty

It's easy to structure a Guide database in question-and-answer format, and in most cases Apple Guide can demonstrate the answers.

People Don't Want to Decide Whether an Instruction Pertains to Their Situation

A Guide database can include context checks so that the help monitors the user's actions and displays the appropriate steps and alerts.

People Don't All Use the Same Vocabulary or Method of Locating Information

A Guide database can present three different ways to find answers. One lets users type their own terms and phrases. The other two (Topics and Index) let users see the tasks and vocabulary.

You can take advantage of these Apple Guide strengths by building them into your instructional design.

Determine the Purpose of Your Guide

Initially, you should determine the purpose of the Guide you're planning. Obviously you want to help Macintosh users do or learn some specific things. Within that context, what's your goal for those users?

For example, do you want your Guide to fill in the blanks, that is, to provide the missing information for someone who is comfortable with the computer but has forgotten (or doesn't know) how to do a particular task? Or do you want your Guide to build a foundation of knowledge for users who are new to the computer or who don't have enough experience to be comfortable using it? Should users be able to find everything about your product or procedure in the Guide, or should they get only the basics?

The purpose a Guide will serve is the primary factor in your decision of which type of database to use. Apple Guide's standard database types can help you make that decision because they mirror the most widely used forms of computer instruction: step-by-step task-oriented help, tutorial, and quick reference. If you aren't certain of your Guide's purpose, you can look at examples of these databases and do a quick survey of how well the subject for your Guide would adapt to each.

Even if you want your database to provide all relevant instructions for all potential users, it's a good idea to establish a primary emphasis and make it your priority as you plan the content. Otherwise, you risk designing a Guide that doesn't adequately meet the needs of any category of users.

Make a Preliminary Choice of Database Type

After you've determined your Guide's purpose, you can make a preliminary decision about the type of database to use. (You indicate the type when you select Preferences in Guide Starter, but you can change the type later if necessary.)

Types of Guides

As mentioned in Chapter 2, Guide Starter offers five types of databases:

- Other (general)
- Help
- Tutorial
- Shortcuts
- About

Other

The Other (general) database is the most versatile because it can be displayed when any program is active. Multiple Guides of the Other type can also appear in the Guide menu, whereas only one of any other type can be listed there. (While the Guide menu can show two or more Other database choices, only one Guide can be displayed on the screen at any one time.)

An Other Guide can be large, with diverse content presented in the Full Access window, or small and concentrated, with only a few tasks in the Single-list format. Examples of when to use this type of database include the following:

- A Guide for all networking tasks that people would perform in a particular workplace
- Instructions for a company's custom applications

- Instructions for logging on to one or more information services
- The procedures for recording and retrieving messages on a voice-mail system
- Step-by-step instructions for an operation that involves two or more programs, such as creating a logo with a graphics program and adding it to slides in a presentation program
- A subset of common tasks from the Guide or the manual for a complicated program

In System 7.5 on the Macintosh, the PowerTalk Guide is an Other database (available only if PowerTalk is installed). Because of its broad usefulness, the Other type is the default choice in Guide Starter.

Help

The Help database is the logical type for complete instructions for a product or a customized program or procedure. Because only one Help database can appear in the Guide menu when a specific program is active, this type is best dedicated to a single program.

Like the Other type, a Help database is well suited to either format in the Access window. In most cases, the Full Access window will allow you the greatest flexibility in presenting Topics to users.

Tutorial

The Tutorial database can ease users into their first computer or first Macintosh and can demonstrate or lead them through basic procedures for a program or a work routine. A tutorial is usually limited in scope, so the Single-list format is preferable for this type of Guide. This format also reinforces another feature of tutorials of all kinds: their sequential presentation of data, which enables a user to learn one skill and then build on that skill in the next lesson.

Although only one Tutorial database can appear in the Guide menu when a program is active, you may want to create a tutorial that's specific to a work group's use of the Macintosh and make sure it's the one available with each program the group regularly uses. That Guide could include some introductory Macintosh skills and some orientation and walk-through for each of the primary activities a new user has to learn in order to be productive in the work group.

As a general rule, Tutorial databases should be able to detect the user's status and have built-in safeguards for the mistakes a novice is

likely to make when following the instructions. However, building in checks and safeguards can require complicated scripting. This is one practical reason for limiting the amount of material covered in a Tutorial database. The most effective way to monitor a user's status and provide for recovery from mistakes is to use context checks, which are explained in Chapter 17.

Another important consideration for a Tutorial Guide is the element of "slow that down and show me again"—what instructional designers call remediation. Tutorials are often designed so that when a user doesn't follow an instruction correctly, the Guide repeats the instruction in a more elementary form. In the Macintosh Tutorial, for example, the Presentation window has a Guide Me button that a user can click to see a simpler version of an instruction. If the user makes an error, the Tutorial detects it and displays a reminder that the Guide Me button is available.

In addition to helping users to obtain basic skills on the computer or on a program, a Tutorial database can do the following:

- Cover essential procedures for using network versions of software
- Offer basic skills and file-handling instructions for students and educators who share computers in a classroom or lab
- Introduce tools and skills practice for high-precision programs, such as fine illustration, CAD, or photo manipulation
- Be a primer on desktop design, using the features of a page-layout program.

Shortcuts

A Shortcuts database is intended to provide quick reference information, such as on keyboard commands or undocumented operations. This type of database can be particularly useful for intermediate and advanced Macintosh users because its content assumes familiarity with the computer and the program or product.

About

An About database offers a brief orientation to the other items in the Guide menu. This database might also include a picture of the Presentation window (like the one used in About Apple Guide in the Macintosh system software).

Other Factors to Consider When Choosing a Database Type

Matching the purpose of your Guide with a database type should be relatively easy. But as you are making this determination, be sure to consider three other factors: size, scope, and diversity of access.

Size

A large amount of material to be included in the Guide can essentially dictate your use of the Help or Other database type. In either case, you will want to use the Full Access window. If coverage will be comprehensive and include tasks that require intermediate or advanced skills as well as basic operations, you'll need the Full format in order to subdivide Topics into enough categories to be useful.

Scope

Similarly, if the Guide will contain a wide variety of Topics, even if the content is not too extensive, you probably will want to build it with the Full Access window.

Diversity of Access

Finally, to give users all three methods of access to the content—Topics, Index, and typed inquiries in the Look For component—you must use a type of database that incorporates the Full Access window.

Your choice of database type is necessarily preliminary at this stage, since you also must assess the audience for your Guide as well as the situation in which the instructions will be used.

Know Your Audience

It's possible to create a well-designed, detailed Guide and then discover its content is not appropriate for the people who will use it. You can avoid such a mismatch (and the extra work it's likely to require to correct) if you first learn as much as you can about the intended users of your database.

Ideally, this research should include determining the experience and confidence level of users and the type of equipment, networks, and shared resources in their environment.

Evaluate the Users of Your Guide

To get the information you need, you can ask questions of one or more representatives of the people for whom you'll create the Guide; for example,

- How long have you used the Macintosh?

  (less than six months; six to twelve months; one to two years; more than two years)

- How many hours per week do you use the Macintosh?

  (one to four hours/week; five to ten hours/week; more than ten hours/week)

- What kinds of work do you do with the computer?

  (word processing; graphics/CAD; page layout; spreadsheet/accounting; database; programming; other)

- Do you share files with others on a network?

  (Yes or No; if yes, do you share (a) files on your computer, (b) files on other computers, (c) network or work group versions of programs?)

- Do you have experience with [the subject of your Guide]?

  (Yes or No; if yes, how long or how many hours/week?)

- How much time per week do you expect to work with [the subject of your Guide]?

  (Estimated hours/week)

This is the kind of data you can assemble in telephone interviews of several members of the group. Or you might prepare a questionnaire that's designed for quick completion and then have a representative circulate it for you. Or if the group has a technical support person or if everyone you talk to suggests that you consult the same expert in the group, you can probably rely on that person's judgment for your assessment of users' experience and needs.

If your research shows a broad range of experience among users, you may want to design separate Guides for inexperienced users and for advanced users. The advanced version would assume basic knowledge and perhaps give many tasks in summary form rather than broken down into individual steps. Or you might design one comprehensive

task-oriented Guide for everyone to use and add a tutorial to bring the inexperienced users up to speed on the computer and the subject area of your database.

Investigate the Computing Environment

Just as you need information about users' computer experience, you also need to know the details of their computing environment, so that you can anticipate their needs most effectively. Again, a technical support person or other expert can usually provide you with this information. You'll also need to find out how well the product your database covers makes use of Apple Guide's features.

Variations in Equipment and Computer Use

The technical demands on your Guide's content depend in part on the diversity of computers and other equipment users have. For example, if some people have Macintosh Plus models or use monochrome monitors, you'll have to provide both color and black-and-white versions of any graphics you use. If some users have access only to small screens (PowerBooks, Mac Plus), you should take extra care to limit the size of the Presentation window by strictly regulating the amount of text and the size of graphics on any panel.

In some cases, you also may need to allow for technical elements that your Guide can't communicate with directly. For example, the task you're documenting may require a user to retrieve data from a non-Macintosh database and then work with that data. If the user connects with the database while using the Macintosh as a terminal, the Apple Guide window probably won't be displayed during that part of the task. Your Guide can begin that task by advising users that Apple Guide won't be visible during data retrieval, but that they can return to the instructions after getting their data.

Compatibility with Apple Guide's Features

You also need to know whether the product or process you're covering is compatible with Apple Guide's features, especially coach marks and context checks. If possible, talk to an engineer or other technical expert to determine if the standard features will be easy to apply in your Guide. If the product can't use essential Guide features, you may need to consult the product's developers about how to make Apple Guide work with it.

The more limitations you have in using Apple Guide's features, the less effective your instructions will be. You may discover that a Guide is not the appropriate way to deliver instructions, for example, if many parts of the product can't display coach marks. (Coach marks are the feature we consider most important in this regard because they literally guide users through a task.)

Of course there will be times when you don't know who the users of a Guide will be or what Macintosh models or monitors they have. In these cases you should plan for as broad a range of user experience and equipment variety as possible.

Do a Task Analysis of the Guide's Subject

After you have gathered the facts about users and the computing environment, you can begin working with the product your Guide will cover. Regardless of the purpose for your Guide, many or most of the instructions you write will lead users through some kind of task. Even if your Guide is a tutorial, you need to know which operations and objectives users must learn. So it's wise to perform (and hard to avoid) a task analysis.

The one exception to this rule may be a reference database, such as a Guide that defines all commands in a program and explains how and when each is used. Yet a task analysis is useful even for a reference Guide, because you'll still need a scheme for organizing the content. You may conclude that the program's menus provide that organization. However, you'll probably want to consider arranging the information by function (task) during your planning.

Task analysis is relatively straightforward: You determine what activities the user can do with the product or procedure for which you're writing instructions.

Organize the Content by Task

A task analysis usually involves three steps:

1. Become thoroughly familiar with the product or other subject for which you'll provide instructions.
2. Determine the major categories of activity for which the product will be used.
3. For each category, determine the tasks users will perform with the product.

In most cases, you can streamline the process of task analysis by checking any notes or documentation for the product or for similar products, consulting the product's developers, and talking with people who frequently use this type of product.

As you determine the tasks for each category, subdivide them into discrete procedures. Be sure to list everything you think a user might want to do with the product. It's good to begin with the most exhaustive list possible. You can trim it later, usually by combining similar tasks.

Also, think about what might go wrong as users perform tasks, and list the troubleshooting information you think they will need. As you develop the task list, make a separate list of terms users will need to understand in order to make effective decisions about the product and its features.

When you've completed the three task analysis steps, you should have something close to the list of Topic Areas and Topics for a Guide, or just the topics, if the number of tasks is small.

Create a checklist for your Guide's coverage of the product by making a double-sided list, one side showing all the tasks (by category) for the database and the other side showing the major features and all primary commands of the product. Number the items in each list so that you can also track which Guide Topics cover which product features and commands. This list will help you plan and revise the Guide's structure. As you write content, you can check off each feature and command your instructions cover. In this way, you can readily see whether you've missed any content and whether coverage is concentrated disproportionately in one or two categories. This information will be very useful as you adjust the categories and add instructions for items that aren't yet covered.

Devise an Alternative Organization, If Appropriate

If a task-oriented presentation won't serve users' needs, you can still use this information to help you determine an effective subdivision of content. As already noted, a command reference arranged by menu organization is one possibility. Other schemes that may be suitable for a reference Guide include a hierarchy of concepts or a glossary or encyclopedia arranged alphabetically.

Complete the Decision Process

After you've completed all the user research and task analysis for creating a Guide database, check your assumptions and solidify your selection of database type.

Finalize Your Choice of Database Type

You can quickly finalize your choice of database type by considering the information you've collected. Be sure to

- review the rationale for your original choice of database type
- ask yourself if anything you've learned about the users' needs, the computing environment, or the product's compatibility with Apple Guide makes your assumptions invalid or prevents your using that type of database; and
- note any concerns or questions you have so that you're sure to address them as you develop the Guide's content.

Determine the Approach and Style for Your Guide

Once you've settled on a database type, consider next the approach and style you'll use in the content. Examples of approaches that work well in Apple Guide include the following:

- The familiar question-and-answer (Q & A) format
- A set of action statements (such as "Write a memo," "Print a manuscript," and so on)
- A command reference or other information-intensive format, or
- A descriptive form of task instructions

Table 4-1 shows an abbreviated example of a descriptive task arrangement for the File Dynamo program.

Table 4-1
Partial content for the File Dynamo program according to a descriptive approach

Topic Area	Header	Topic
File-management strategies	Reducing clutter	with folders with archives by selective erasing
	Saving disk space	with file compression with aliases [... (etc.)]*
	Navigating quickly	in list view with the Find command with aliases
	Backing up	current work essential files [... (etc.)]
Files	Understanding	file attributes aliases
	[... (etc.)]	[... (etc.)]
Copying, moving, or using aliases	Automating	software installation file backup server cleanup
	[... (etc.)]	[... (etc.)]
Renaming	Automating	file locating date in file name prefix or suffix in file name
	[... (etc.)]	[... (etc.)]
Setting Attributes	Automating	program memory adjustment file type change
	[... (etc.)]	[... (etc.)]

* [... (etc.)] indicates additional entries

Once you've decided on the approach, go through your list of tasks and categories and ensure that all of them can be presented logically and clearly with that approach. If you have difficulty finding words and phrases that you're confident users will grasp quickly and find useful, try another approach.

If your content is fairly complex, devise the task list in two different ways—as Q & A and as descriptions, for example—and then compare them, choosing the one that better fits the product covered by the Guide. Consulting someone who's familiar with the product about your initial list of tasks is always useful as well.

Prioritize the Tasks and Other Information

Usually your Guide-development project must meet a schedule and budget, either of which may require that you limit the content in the Guide. As you plan the tasks and information, it's a good idea to prioritize the areas of coverage so you can complete the most needed instructions even if the project ends prematurely.

Keep the following considerations in mind as you determine the priority of content for your Guide:

- If you have access to a customer support group for the product your Guide covers (or a closely related product), find out the ten or twenty questions the support staff answers most frequently. Then provide full instructions and explanations for the problems those questions identify.
- Cover the most commonly used features or tasks, especially ones that are complex.
- Provide recovery information for any tasks or situations in which users might lose their work if they make a wrong choice.
- If your Guide is the only documentation that will be provided with the product, make it as thorough and comprehensive as you can.
- If other reference material is provided with the product, you may be able to limit your Guide's coverage to primary procedures and tasks.
- If the audience for your Guide is largely experienced users, include tips, shortcuts, and as much power-user information as you can.
- If the Guide is designed for PowerBooks or other portable devices (such as a modem) concentrate on tasks that users need to do when traveling. Emphasize efficiency of use and provide strategies to conserve battery power.

Design for the Unique Demands of Online Instruction

It may seem late in the design process to introduce more considerations at this stage. But you have to be this far along in your Guide planning (knowing the type of database and having an initial list of categories and tasks) before you can apply the unique requirements of online instruction to your database.

The following characteristics of users' responses to online help (or in one case, a limitation of the medium) apply to every database, regardless of its content or organization. The way in which you implement the content and organization, however, can do a lot to compensate for the disadvantages inherent in paging through a series of boxes on the screen.

Users Quickly Develop a Set of Expectations

Once users have gone through the instructions for two or three tasks and used the Topics listing to locate information, they will have developed a set of expectations for how your Guide behaves. For example, if the steps for choosing a printer lead users through each part of the Chooser, displaying coach marks at all action locations, users will expect coach marks on most or all other parts of the interface that they click on or drag across. If you use coach marks only in some tasks or only on some part of the interface, users won't be as confident of your instructions.

Users Won't Read Much Text on the Screen

Four years of usability testing with Apple Guide prototypes demonstrated convincingly that users are reluctant to read much text on the screen, especially in preparation for taking an action. Many users are so impatient with reading instructions that they'll spend twice as much time accomplishing a task by trial and error as it would have taken them to read and follow the steps in the online help.

Users Can't See the Whole Thing at Once

One advantage of online help is that it's always available when you're using the computer. One disadvantage is that it's not a book. So while users can consult online help almost at will, they can't see it all at once, gauge the complexity of a task by counting the pages for it, or flip through it until they see something familiar.

One way to compensate for the lack of a broad perspective is to provide a sort of blanket coverage by listing tasks in every category that's relevant to them. For example, if your database covers compact discs, be sure to list them under both "Disks" and "Sound."

Users Can't Use Online Help When the Computer Isn't Operating

Obvious as this fact may seem, it's a good idea to check your list of tasks for content that assumes computer use when such use may not be practical. You don't necessarily have to eliminate these tasks from your Guide, but you do need to verify that they are covered in a manual or other printed material.

Among the instructions that users may not have available if the computer is not functioning are the following:

- Hardware setup, installation of expansion cards or other components inside the computer, or connection of peripherals to ports on the computer
- Turning the computer on and off
- Formatting or partitioning of hard disks (these instructions are useful in a Guide as well)
- Safety and maintenance information
- Troubleshooting information related to hardware or system crash
- Orientation to Apple Guide and the Guide (Help) menu

Refine Your Design

Go over your initial task list with the special online considerations in mind—users' expectations, impatience, and limited view. This is a good time to make notes about how your Guide will establish a consistent approach to tasks, how you can write instructions that give the essentials up front for impatient users, and how to give users some idea of what they aren't able to see.

The strategies you develop as you plan the Guide's organization will be useful as you write the instructions, so keep them close at hand. When you're feeling buried by the details, it's often helpful to be reminded of the ideas and objectives you had at the beginning of a project.

The design process is complete when you've written down the plan for your Guide, either on paper, if that's easy or a natural way for you to start, or perhaps in the outliner of a word processing program. At this point, you don't have to list every bit of content that will go into your database, but you should have a working draft of the following items:

- The Topic Areas or other major categories of information in the database
- The Topics in each major category
- The parts of the product's and computer's interface that you want to indicate with coach marks
- The likely problem areas for users—either potential difficulties inherent in the product or tricky parts of the instructions

Once you know this much about your Guide, you're practically finished. All you have to do now is open Guide Starter and start writing.

FIVE

WRITING CONTENT FOR THE ACCESS WINDOW

Now you can start building your Guide. If you haven't already done so, copy the software from the Starter Kit disk to your hard disk (see "Installing the Starter Kit Software" in Chapter 2 for instructions).

Hands On: Entering Database Details

Initially, you begin working in Guide Starter. After you set the Preferences, Guide Starter makes a copy of the program with the settings you've chosen and the name you supply.

Follow these steps to begin using the program:

1. Locate the program named Apple Guide Starter (inside the Apple Guide Starter Kit folder). Leave the program inside the folder.
2. Open Apple Guide Starter.

   You'll see a welcome message explaining that the program needs some information from you. As the Welcome text also notes, Guide Starter creates a new application that contains the attributes you specify in the Preferences screens.
3. Click Continue to see the first of the four Preferences screens.

Select the Type of Guide

The first attribute you specify is the type of Guide. Refer to the following descriptions if you haven't yet decided which type is appropriate for the product your instructions will cover.

Click the button for the database type you want. Your choices are the following:

- Other (general)—like Help, generally uses Full format; can be task-oriented or reference style; can be large or limited in scope; multiple Other databases can appear in the Guide menu
- Help—uses the Full format for Topic Areas and Topics; usually task-oriented; accommodates a large amount of information; only one Help database can appear in the Guide menu when a specific program is open
- Tutorial—uses Single-list format for Topics; usually presents Topics in a sequential order, with the content of each section building on the previous one; often designed to contain lots of remediation and repetition
- Shortcuts—short items, often in a visual format such as showing icons or keys; may not need the Access window
- About—a brief database to explain the other Guides, ideally providing visual orientation to using help; usually doesn't need the Access window

(See "Make a Preliminary Choice of Database Type" in Chapter 4 for a comprehensive discussion of Guide types and their uses.)

Select the Access Window Format

Once you've chosen a database type, specify either the Full format or the Single-list format for the Access window. (The Tutorial and About databases can only be in Single-list format in Guide Starter.)

1. Click the Full Access or Single List button to select a format for the Access window.
2. Click Continue to move ahead.

Full Format

For most databases, the Full format is the appropriate choice because it provides a two-level subdivision of content. Hence your Guide can cover a large number of tasks, and users can see as many as 132 Topics without having to scroll in either list.

The Full format is also quite flexible. If necessary, you can have a widely varying range of content in the Topic Areas. This means that your Guide can offer instructions for procedures that encompass several different programs or sets of unrelated activities that everyone in a work group or a classroom may need to use.

As Figure 5-1 shows, the Full Access window also provides three ways for users to locate Topics. In addition to the Topics list, Guide Starter can create an Index and assign Topics to its terms. You also can create a thesaurus of words and phrases to match text that users type in the Look For component. (You add Look For content in the text file for a Guide; see Chapter 15 for details.)

Single-list Format

As noted previously, the Single-list format is excellent for linear sequences such as a Tutorial or for a database that has a limited number of tasks. However, the format doesn't offer an Index or Look For

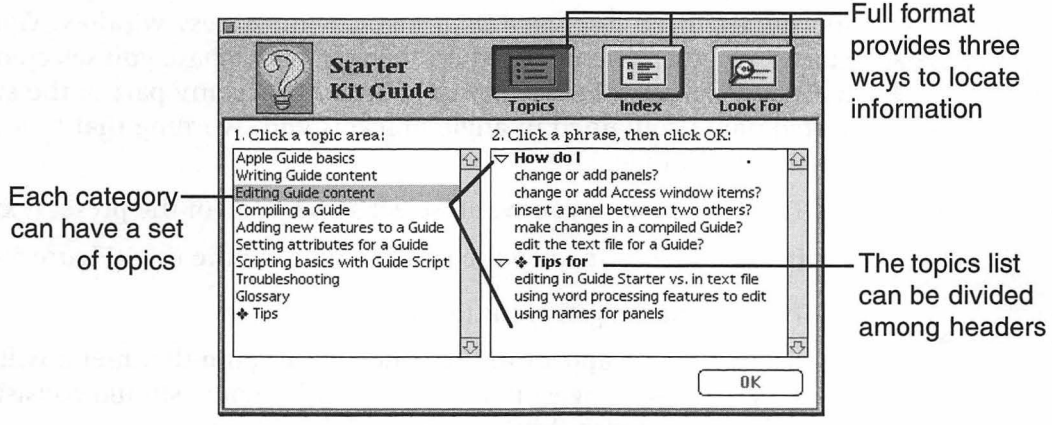


Figure 5-1
The Full Access window showing two lists and three access methods

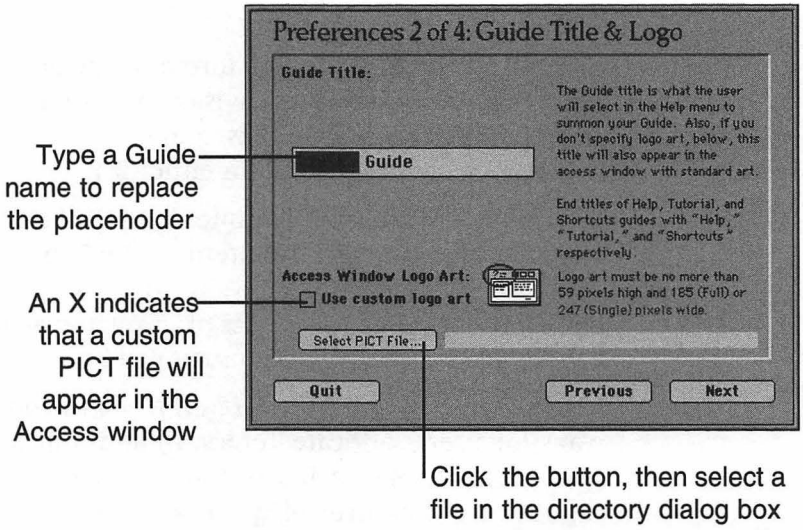


Figure 5-2
The name and logo art options in Guide Starter's Preferences

component, thereby making it less useful for a database that contains a large amount of information or a sizable number of specialized terms and concepts.

Name the Guide

The second Preferences screen lets you name your Guide and specify art to be displayed as a logo in the Access window. Guide Starter suggests a name based on the type of database you selected in the first Preferences screen. You should replace any part of the suggested name contained in angle brackets with wording that best describes your Guide.

1. To add the name you want, select some or all of the preset text.

   It's a good idea to keep the word "Guide" in the title (Figure 5-2).

2. Type the name you want to use.

   The name will appear in the Guide menu (and that menu will be as wide as its longest item), so the entire name should consist of only two or three short words.

> **Tip** It's a good idea to write down the name you gave the Guide, along with a summary of its content and the date (and update your notes if you change the name later). Keeping a digest of the Guides you create gives you a quick reference source and saves time that you might otherwise have to spend looking for and opening Guides or text files that have similar names or content. If you establish such a record, include the status of the database each time you add to or revise it and each time you compile. Then if you have a problem compiling a Guide, you can go back to your notes and determine at what point the error occurred.

Add Logo Art

If you want to use your own logo or other graphic in the Access window of your Guide, you can indicate to Guide Starter what file contains the art, as follows:

1. Click the checkbox labeled "Use custom logo art" to put an X in it.
2. Click the Select PICT File button.
3. Use the directory dialog box to locate the logo or graphic file. Select the file and then click Open.

Use a PICT File

The graphic you use for the Access window must be in the PICT format. This format is the type of Macintosh graphics file usually created by a draw program or any of the numerous graphics programs that create files in several formats. The graphic also must fit the available area in the Access window, which measures 59 pixels high by 185 pixels wide for the Full window and 59 by 247 pixels for the Single-list window. (Pixels are pictures elements, the tiny squares that make up a monitor's screen. Most graphics programs that create PICT files can show an image's size in pixels.)

In the Apple Guide Starter Kit folder are blank templates you can use to create your own custom logo art. These templates consist of the marbled background graphic that will best blend with either size

of Access window you choose. Because the Single-list Access window has no buttons across the top, its art space is wider (247 pixels) than the Full window's (185 pixels). Create your custom art and paste it on the template and then save the file with a new name. Be sure to include the Guide's name in the art.

Important: When you use one of the custom logo templates or create your own logo from scratch, place the art at the upper left-hand corner of the window before you save it. Otherwise, the logo will not be in the correct position when the Guide is compiled.

Figure 5-3 shows a sample PICT with a logo and Guide title. This file is the size for the Full Access window.

> **Tip** If you're working on several Guide files or you have trouble remembering what you named a Guide, you can ensure that you'll always know the database name by waiting until just before compiling to tell Guide Starter to use a custom logo file. If you don't select custom art, your Guide Starter application displays the Guide's menu title as text in the Access window. If you do select custom art, Guide Starter displays the message "Your Logo Art Here."

Include the Guide Name in Logo Art

In addition to having a graphic or logo, the PICT file must include the name of your Guide. Apple Guide can import your logo file or display the Guide Title you supply, but it can't combine these two elements for you.

Figure 5-3
The Starter Kit's sample logo art for a Full Access window

> **Tip** If you have a big database that takes a fair amount of time to compile, make a small database with one task of dummy content to try out your logo. You can then compile this small logo database, evaluate it, adjust the logo, and recompile. Repeat this cycle until you get it just right. As with all experimental files, change the file name—do Save As—for each new Guide file so that you can go back and retrieve any of them you want. Insert a version number between two of the first few words of the file name so that even with long file names, you'll be able to identify each file in Finder list view.

Important: To avoid file-location problems when you compile or recompile a Guide, be sure to keep all files Guide Starter needs in the Apple Guide Starter Kit folder. When you specify a PICT file, the application notes the path to that file. If you move the PICT and later compile a Guide, the compile procedure may fail if the program can't locate the PICT file.

Click Continue to move ahead.

Write Greeting Text

The third Preferences screen is where you enter the Welcome message—the "Howdy text"—that is displayed the first time a user opens the Guide.

1. Delete any of the preset text you don't want to use.
2. Type the new text you want users to see when the Guide opens.
3. Click Continue to move ahead.

The preset greeting for the Full Access window explains the purpose of the Topics, Index, and Look For buttons. (The Look For button is dimmed in Guide Starter, but it always appears in the Full Access window of a compiled Guide. You can add Look For content to the text file, as explained in Chapter 15.)

This text should advise users of the primary ways they can begin using the Guide's content. The text should also be as brief as you can make it because it competes with several other elements in the Access window that will distract the already reading-reluctant users.

Specify Location and File Information

Guide Starter stores a path that it uses to place the finished Guide in a folder you specify. There are rules for locating a database according to what program or programs it provides instructions for. Follow these guidelines for setting a Guide location:

- If the Guide should be in the Guide menu when a specific program is active, the database must be in the same folder as the program.
- If the Guide covers more than one program, you can duplicate it and store a copy in the folder with each program, or you can put an alias of the database in each program's folder.
- If the Guide is designed to work with a specific file or set of files, the database must be in the folder with the program for each of those files. (If the database covers files from several programs, put the Guide in one program's folder and put aliases or copies of it in the other programs' folders.)

  Some programs put their own help files in the Guide menu, so they may not recognize or display an Apple Guide file. In this case, try changing the database to a different type (in a copy of the text file) and then recompile to see if the new version appears in the Guide menu.

- If the Guide should be available when the Finder is active, you need to put it in the Extensions folder inside the System Folder of your hard disk. (Note that you can't just drag the file to the System Folder and expect the Macintosh to put it in the Extensions folder, as you do with many items. You must open the System Folder and be sure to put the Guide in the Extensions folder.)

  Important: If you put a Guide in the Extensions folder and it is any type except Other, that Guide will replace the one of its type supplied by Apple. If two databases of the same type are both available for an active program, Apple Guide displays the one created most recently.

- If you want the Guide available at all times, make it an Other database. Then put copies or aliases in the folder for each program, a copy or the original in the System Folder, and a copy at the root level of the hard disk in your Macintosh. (As already noted, some programs that display their own help in the Guide menu won't recognize an Apple Guide database. Try changing your Guide to another database type if this happens.)

➤ Remember that only one Help, Tutorial, Shortcuts, and About database can be displayed in the Guide menu. The name of the most recent one will replace the name of an older one of the same type. Multiple Other databases can appear in the Guide menu.

Specify Location

✔ With these considerations in mind, select the primary location for your Guide. You can make copies or aliases later if you want the database to appear with more than one program.

1. Click the Select Folder for Database button.
2. Use the directory dialog box to locate the folder or disk where you want Guide Starter to put your database. Then click Choose (below the file list) to designate the location.

If you don't select a specific location for your database, Guide Starter will insert the path to the folder containing your Guide Starter application the first time you compile the Guide.

Specify Associated Program

✔ If your Guide covers one program, you can ensure that program is opened each time you use the Guide Starter application for which you're setting attributes. Having the program open is always useful—and usually necessary—so that you can write or revise instructions for it and specify coach marks for the program's window features. (If the Guide covers a file, you must specify the program that created the file.)

1. Click to place an X in the checkbox labeled "Always Launch Application While Editing Guide" if you want the program active whenever you open the Guide Starter application for the Guide.

   Because of the memory requirements of System 7.5, Guide Starter, and Guide Maker Lite, you should not check this box if your Macintosh has only 8 MB of RAM. There won't be enough memory available to compile a Guide. (You can work around this limitation by writing content with the application open and quitting it before compiling.)

2. Use the directory dialog box to locate the folder or disk where the associated program is located. Then click Open to specify the program.

Regardless of whether you want the program opened when you reopen the Guide Starter application, you can specify the name and location of the program for which the Guide provides instructions. (You don't have to specify a program, however.)

Important: If you check the "Always Launch" box, the associated program will open every time you open your Guide Starter application. Once you've compiled a new Guide, you must quit the associated program and reopen it for the program to recognize your Guide. As long as you don't change the Guide type or name, you may recompile revisions to your Guide and see the changes without restarting the associated program.

Supply Version Information for Your Guide

✔ You and others who use your Guide will need to know which version of the instructions a particular copy of the database represents.

1. Click in the first text box and type the Guide name and version number.
2. Click in the second text box, then type the version number and copyright information.
3. Click in the third box and type the version number.

Figure 5-4 shows the Info window for the first draft (alpha 1) of the Guide for Guide Starter.

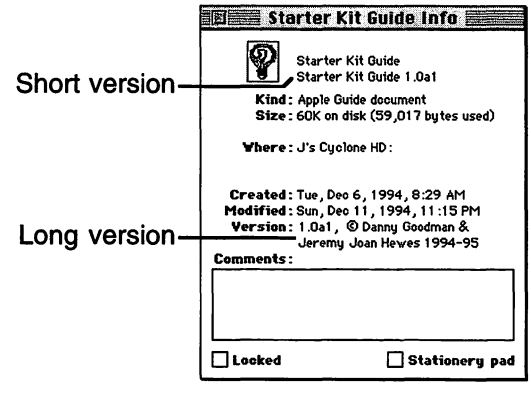


Figure 5-4
The Info window for the first draft version (alpha 1, or a1) of a Guide

The version number needn't reflect every time you revise (although it can if you wish). Usually the version number indicates the stage of development of the product or a release number to help you track bugs and user problems.

For a database, you should change the version number at least when you add to the database's content substantially or make major changes in its organization or instructional approach. (You can use the names and dates of your text files to monitor minor revisions.)

Review Your Choices and Name Your Application

You may want to review the attributes you selected and the title and other information you supplied. Use the Previous button to go through the Preferences screens in reverse.

1. When you've reviewed the information and made any changes you want, click the Next button to move forward through the screens and click Done on the fourth screen.
2. Use the directory dialog box to select a location for the application that Guide Starter will create. Then type a name for the application and click Save.

Guide Starter will close, and your new application will open with the Access Window Builder. This new application is the one you launch the next time you want to revise or add to this Guide. Open Guide Starter again only when you want to start a new database.

> **Tip** You can go back to the Preferences screens to review or make changes any time you're using the application by choosing Preferences from the Edit menu.

Enter Guide Topics

The Access Window Builder should be waiting, so you can begin entering your categories (Topic Areas, in Full format) and Topics. As you work, you needn't be concerned about saving the information you've entered—Guide Starter saves automatically.

> **Tip** Whenever you're working with a Guide Starter application, you can use the Starter Kit Guide for step-by-step instructions. (Because Apple Guide can display only one Guide at any time, the Starter Kit Guide will replace the Guide on the screen, if one is open.)

Headers and Topics: Questions or Statements?

The wording you use for Headers and Topics in Apple Guide Starter influences the wording that appears at the top of all panels in the Presentation window. Apple Guide Starter adheres to the following convention:

- When a Topic ends with a question mark, Guide Starter displays the combined Header and Topic at the top of each panel in a sequence.
- When a Topic does not end with a question mark, only the Topic appears at the top of each panel.

For example, if a Header reads "How do I" and one of its Topics reads "save a file?" then the panel title for all panels in that task reads "How do I save a file?" But if the Header reads "Terminology" and one of its Topics is "color wheel," the panel title will be simply "color wheel." Therefore choose your wording and capitalization for Headers and Topics in anticipation of what users will see while navigating through the Guide.

Full Access Window

Follow these steps to enter content in the Full Access Window Builder:

1. If necessary, make your Guide Starter application the active program, with the Access Window Builder the open window.
2. Click the Edit button for the Topic Areas list (on the left).

   An extension opens at the bottom of the window. This is where you enter text and edit the list (Figure 5-5).

3. Click in the text box at the bottom center of the window. Then type the first Topic Area name and click Add.

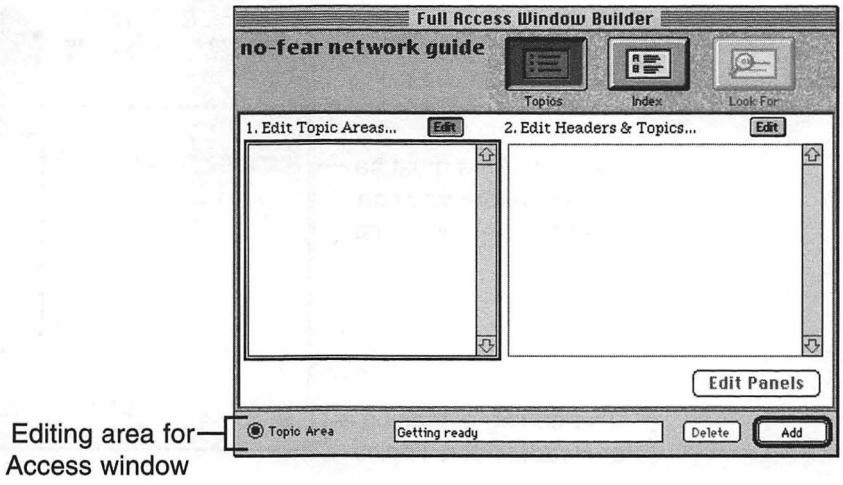


Figure 5-5
The Full Access Window Builder with the editing area in view

The text you typed appears in the Topic Areas list with a number in brackets after it. Ignore this number; Guide Starter uses it to manage the information in your Guide.

4. Continue to enter Topic Area names until all are listed.

   If you are a fast typist, you may notice that the first few letters don't appear in the text box. They will show up in the Access window list, however.

5. Select the Topic Area for which you want to enter tasks.
6. Click the Edit button for the Headers & Topics list (on the right).

   The editing extension opens at the bottom of the window. This one has two radio buttons so you can switch between Headers and Topics (Figure 5-6).

   The Header button is selected the first time you begin editing this list (unless you have previously selected a Topic already listed).

7. Click in the text box at the bottom center of the window. Then type the first Header name and click Add.

   When your new Header is displayed, the Topic button is highlighted automatically.

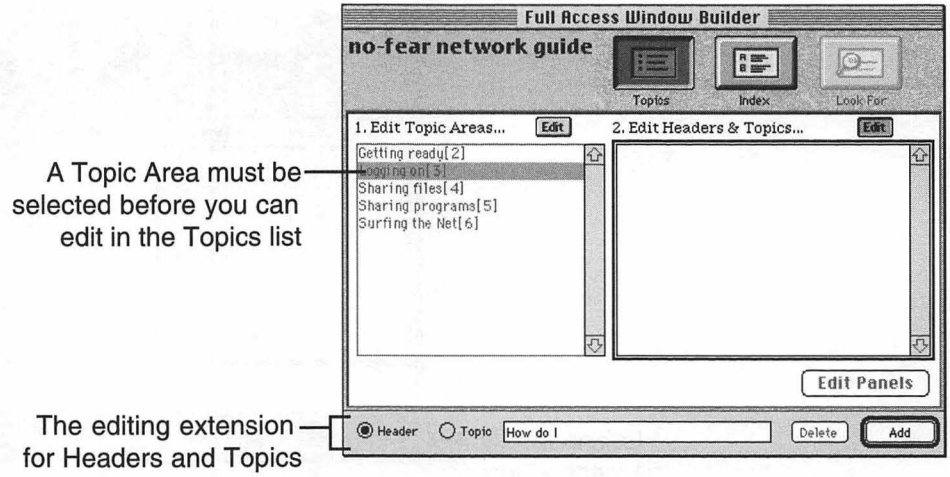


Figure 5-6
The Full Access Window Builder with editing open for Headers and Topics

8. Click in the text box at the bottom center of the window. Then type the first Topic name and click Add.
9. Continue to add Topics until you've added all for the Header. Then switch to the Header button and repeat the procedure.

You can finish entering all Topic Areas, Headers, and Topics now or try entering instructions for one or more Topics. You can always return to the Access Window Builder to add or edit its lists (by clicking the Edit button for that list).

Important: Although it seems logical to add Index terms early in the Guide-authoring process, wait to begin the Index until you've written instructions for most or all of the Topics. Guide Starter is designed so that you can link Index terms to the Topics for which you've entered content (that is, panels in addition to the Access window lists).

If you plan to include definitions of key terms in the Topics section of the Access window, enter the Header "Definitions" and list each term as a Topic. (If you aren't certain whether to include definitions, look at Macintosh Guide for examples of definitions listed, along with tasks.)

Single-list Access Window

Entering Headers and Topics in the Single-list Access Window Builder is a simple procedure. Follow these steps to create a Topic list.

1. Click the Edit button above the list at the upper right-hand corner.

   The editing extension opens at the bottom of the window. It has two radio buttons so that you can switch between Headers and Topics (Figure 5-7).

   The Header button is selected the first time you begin editing this list (unless you have previously selected a Topic already listed).

2. Click in the text box at the bottom center of the window. Then type the first Header name and click Add.

   If you are a fast typist, you may notice that the first few letters don't appear in the text box. They will show up in the Access window list, however.

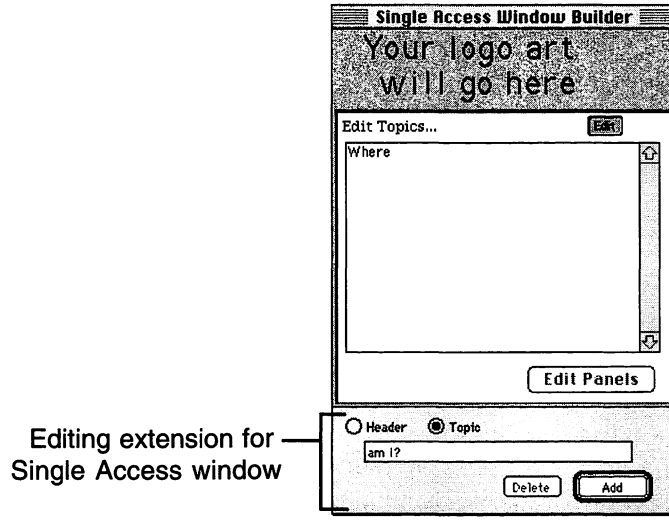


Figure 5-7
The Single-list Access Window Builder with editing open for Headers and Topics

3. Click in the text box at the bottom center of the window. Then type the first Topic Area name and click Add.

   The text you typed appears in the Headers & Topics list with a number in brackets after it. Ignore this number; Guide Starter uses it to manage the information in your Guide.

   When your new Header is displayed, the Topic button is highlighted automatically.

4. Click in the text box at the bottom center of the window. Then type the first Topic Area name and click Add.
5. Continue to add Topics until you've added all for the Header. Then switch to the Header button and repeat the procedure.

You can finish entering all Headers and Topics now or try entering instructions for one or more Topics. You can always return to the Access Window Builder to add or edit its lists.

Review the Topic Phrases for Clarity

The information you enter in the Access window is the framework for your Guide. Users will find the information they need relatively easily if the Topic Areas and Topics are logical and the vocabulary and phrasing are direct and contain minimal jargon. If the terminology is technical and the users aren't, the Guide's instructions may not get much traffic, no matter how thorough and helpful they are.

As you work with the list of Topic Areas, Headers, and Topics that you've planned for the Guide, try to look at each item through the eyes of a perplexed user. Also, try or to capture the language that people around you in the workplace use to describe their use of the computer. Another tactic that's often helpful is to ask a variety of users how they would describe the Topics your Guide will cover.

Access Window Text Limits

Obviously your Topics must also be succinct. In the Full Access window, you have all of 29 characters for a Topic Area and 37 characters for a Topic (not including the Header). In the Single-list format, Topics can be 37 characters. These are approximations, however, as the actual number varies according to the letters used.

As you gain experience with Apple Guide, you'll likely become adept at structuring information and writing task descriptions within the boundaries imposed by the Access window. Until you're adept, expect to do some trial and error.

Consistent Style

Another aspect of Guide authoring that improves with practice is a consistent presentation style of tasks within Topic Areas. For example, if a program has four tools palettes, you could present each palette along with a description of each tool and a brief list of the types of tasks it is used for. An alternative arrangement might be to list all the tasks performed with each tool. What you don't want to do is mix those two species of organization so that some palettes are presented tool by tool and some tools are discussed task by task. That's a mongrel database, not a trusty Guide dog.

Review the Topics for Scope and Efficiency

When you've entered your Topic Areas, Headers, and Topics, be sure to look closely at each list to determine

- that the tasks are separate, distinct activities and that none overlap parts of other tasks, and
- that the order of items is the most appropriate for your Guide's content and users and the most efficient in covering the subject.

If you see that some tasks can be combined or that items in a list would benefit from a different arrangement, make the changes before you begin writing Topic content.

Combine Similar or Overlapping Tasks

Depending on the subject and size of your Guide, you may want to combine tasks that are similar into a single task that integrates their content, either to save space or to make the task more logical for users.

Saving Space

You should combine tasks if a Topic Area has many tasks and you'd like users to see all those tasks without having to scroll through the list in the Access window. For example, you could create a general task and present several related subtasks within it. Table 5-1 presents several examples of subtasks combined into a larger task.

Table 5-1
Examples of small tasks combined into a single large task (from Macintosh Guide)

Combined Task	Subtasks
How do I print?	How do I select a printer? How do I use a printer on a network? How do I use a serial printer? How do I print a document from a program? How do I print a list of files on a disk? How do I print a copy of the desktop?
How do I prepare a disk for use?	How do I initialize a floppy disk? How do I initialize a hard disk? How do I partition a hard disk? How do I format a DOS disk?
How do I adjust the way the keyboard works?	How do I adjust the repeat rate or delay before repeat? How do I make keyboard commands easier to type? How do I set the keyboard to ignore accidental keystrokes? How do I control the pointer with the keyboard?
How do I make an item easy to find?	How do I make an alias? How do I put an item on the desktop?

Improving Task Logic

Another good reason to combine two or more small tasks is to create a more logical Topic for users. The final item in Table 5-1—"How do I make an item easy to find?"—is an excellent example of this approach. By itself, the subtask "How do I make an alias?" is not particularly informative to users because it gives no clue about what an alias is or what it might be used for. When this subtask is nested within the "easy to find" task, however, users can readily see the logic of using an alias. Hence they are more likely to use the instructions than those for a "make an alias" task.

To make the most efficient use of large tasks that combine several smaller ones, you can use one or more of the advanced Apple Guide features that you add to the text file for your Guide. See "Be Familiar with Advanced Features" in Chapter 6 for an explanation of different ways to structure a task.

Rearrange Items as Necessary

As you decided on Topics and later entered their descriptive phrases, you may not have seen them all together or determined the best order. Take a few moments now to consider the lists and move items around to see how an alternative arrangement works.

To rearrange items in the Access Window Builder, follow these steps:

1. Click the Edit button above the list you want to rearrange.
2. Drag the item you want to move to a new position (Figure 5-8).

   You can move any type of item—Topic Area, Header, or Topic. Again, ignore the numbers in brackets after items, even if they appear to be out of order or a number is missing after you delete something. Guide Starter uses the numbers to manage its data.

Consider arranging the items in the Access window according to priority or frequency of use, order of use, or relative simplicity and complexity. Or you can let the alphabet determine the order.

Priority Order

One approach to listing is to arrange Topics in order of use, with those likely to be used most often at the beginning of the list and those to be used least often at the end. Figure 5-9 shows a list of tasks

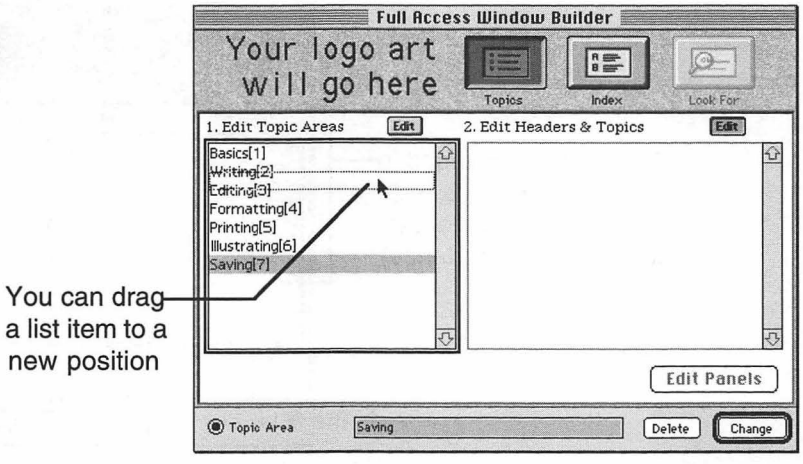


Figure 5-8
Pointer and outline show item being moved to new position in list

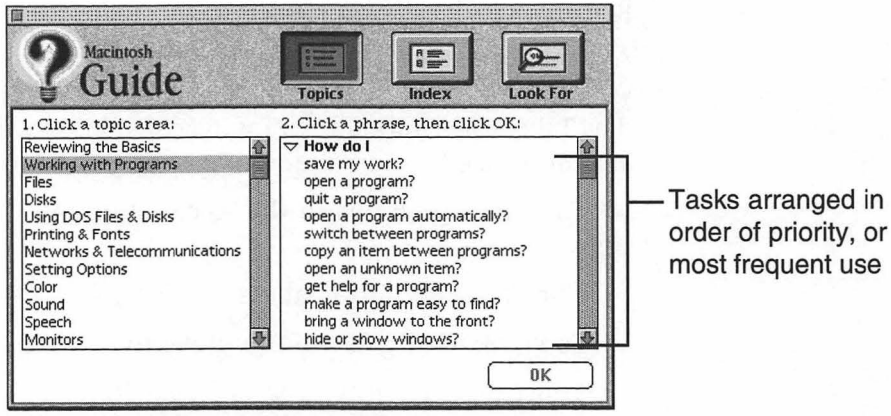


Figure 5-9
Macintosh Guide tasks arranged in order of most frequent likely use

in Macintosh Guide arranged so that the ones users are likely to choose most often appear at the beginning.

Operational Order

Another approach to listing is to place Topics in the sequence in which a user would use the instructions and information when going through the process covered in the database. Figure 5-10 shows an example of this arrangement.

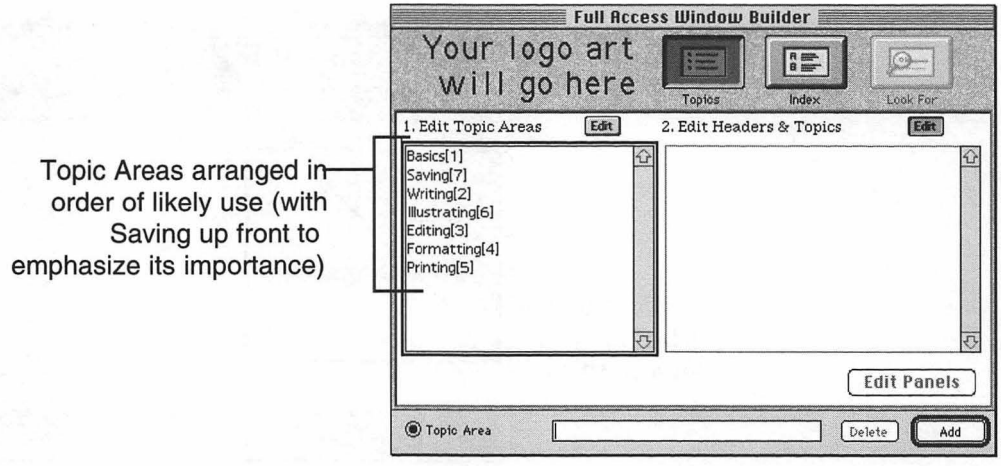


Figure 5-10
Topic Areas arranged in the order they occur in the publishing process

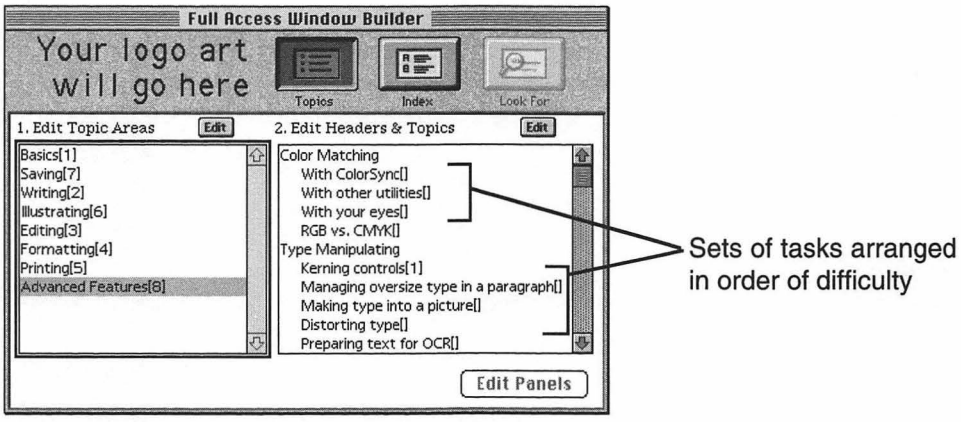


Figure 5-11
Two series of tasks listed in order of difficulty, with simplest first

Order of Difficulty

You can also order tasks from the simplest to the most difficult or most complicated. Figure 5-11 shows two brief examples of this approach to listing.

Alphabetical Order

Listing items in alphabetical order is necessary for the Index of a Guide database and is usually the best arrangement for definitions in a list. Another logical application of alphabetical order might be for a large reference database intended for use by people already familiar with much of the terminology it contains.

Continue rearranging the items in the Access window until they are in the order you want. You can always change these lists later as you develop other parts of your Guide. When your Topic lists are complete, turn to Chapter 6 to begin working on instructions.

SIX
PLANNING PANEL CONTENT

Now that you've designed your Guide's structure and listed some or all of the Topics in the Access window, you're ready to load the ammunition. Well, not quite, but you will be leading users through the steps of each task as you write the panel content. And because Apple Guide is interactive, you must always be aware of the potential for users to take a wrong turn or wander off the path.

With careful planning, you can provide useful, economical instructions that keep users moving ahead in their work without getting lost or giving up.

Hands On: Determining the Details

This chapter focuses on planning content for the panels, so the hands-on work involves using the product that your Guide covers. These activities include

- using the product the Guide covers to do each task and
- mapping the panels in each Topic (on paper or on the computer).

Plan Panels for Topics

The key to developing such useful instructions efficiently is determining the content and sequence for all panels in each Topic before beginning to write the text. You'll get the best results by doing some "homework" and then applying some criteria as you make decisions for each Topic and panel.

Learn from Good Examples

A bit of advice we can't give too often is to look closely at Guide files that provide instructions similar to the ones you are writing. The databases integrated into Macintosh System 7.5 offer examples of most Apple Guide features and instructional approaches and represent Apple's recommended way of creating online help. Look for other Guides among commercial applications and shareware programs and utilities.

Do the Tasks

If you've written instructions before, you know how easy it is to overlook a step or a key detail in a task. So it's important that you do each task as part of preparing to include it in your Guide. As you go through the tasks, try to make your work situation as much like that of the Guide's users as possible.

Map Each Sequence

Actually going through the steps of each task gives you the breakdown and sequence of actions for a task. For maximum accuracy and efficiency, write down this information, along with any tips or potential problems for users that you notice while doing the steps. (If your Guide has Topics that don't involve actions—and thus are not tasks—you should still map the sequence of panels for them to ensure you include the necessary information.)

Also consider using a visual design aid that will both standardize and accelerate the planning process for organizing Topics in your Guide. You can diagram the sequence of panels in a Topic simply, such as by sketching boxes and labeling them on paper, or more elaborately, such as by assembling a series of objects in a computer file to represent the panels in a Topic.

If you're familiar with programming, you may have already designed flow charts, which represent different parts of a program or process with different shapes. As part of a database flow chart, you'll want to include the following:

Basic Features

- action panels
- information panels
- coach marks
- Huh? buttons

Advanced Features (Added in the Text File)

- radio button/branching panels (see Chapter 14)
- AppleScript elements (see in Chapter 16)
- context checks (see Chapter 17)

Develop a Standard Structure for Topics

Another way to streamline the work of writing instructions and to assure uniformity of content in a Guide is to work out a standard structure for Topics. Although you'll have to put some effort into devising this structure, it should serve as the basis for any Guides you write in the future.

Establish a Set of Standard Panel Types

Almost every Topic has certain common elements, such as an introduction that summarizes the activity, the definition of one or more key terms, and a final action (such as closing a window) that completes the task. By establishing standard panels to represent these repeated elements, you can quickly decide whether one of the standard panels applies to an action or explanation at each point in a sequence.

At the most basic level, standard panel types include the following:

Primary

- Action panels, which tell users to do something
- Information panels, which explain a procedure or concept

Secondary

- Decision panels, which contain radio buttons for users to choose a branch in the Topic (see Chapter 14)
- Advisory panels, which alert users that they did a step incorrectly (see Chapters 16 and 17)
- Cross-reference panels, which direct users to tasks that are closely related to the one they're doing
- Tips, which provide helpful information about the current action or the task itself.

As you gain experience at designing and writing Guide instructions, you'll probably develop additional types of standard panels that suit the needs of your Guide's subject and users.

Make a Panel Template

You may want to fashion a template that you fill out for each panel, noting its format, coach mark, Huh? information, and so on. If the Guide you're creating is large and complex, use a database program for the template so that you can sort its contents to list all panels with a certain attribute or feature, such as a Huh? button that opens another task.

Or a template could simply list all the items that can be associated with a panel. Each time you start a new panel, you fill in a copy of the template. You could maintain this information in a word processing program (and use the program's search capability to locate all instances of a certain feature, if you use a standard set of terms). Or you could print copies and fill them out by hand.

Maintaining such a full set of information for each panel is probably not necessary once you've completed the writing. But a minimal version, such as the checklist in Table 6-1, is quite useful for making decisions about a panel as you're planning the instructions for a task.

Answer a Series of Questions for Each Panel

Whether or not you use a template to record the decisions you make about each panel, you can determine the content for the panel by asking yourself a series of questions about it. Your answers to questions such as those that follow will shape the text you write for the panel and the panel options—such as coach mark, Huh? button, and prompt text—you specify.

Table 6-1
Example of a template to help plan and track information for each panel

Panel Attribute	Style Used (Circle One or Fill In)	
Format	Full Tag	Radio button* Other*
Coach mark—location	None Menu Dialog item*	Window—standard Window—rect Desktop*
Coach mark—type and color	None Circle (red/_____)* Menu (underline)	X (green/_____)* Arrow (red/_____)*
Huh? button	None AppleScript*	Graphic*
Prompt	None Standard	Custom
Context check*	None Apple Guide standard	Custom module
Auto-open AppleScript*	None Menu item	Window Other
Panel used elsewhere?	Yes	No
Panel name (optional)†	_____	
Panel number (assigned by Guide Starter in text file)	_____	

*Details provided in Part III, "Adding Bells and Whistles."
† See Chapter 11 for information about naming panels.

Of course not all of the following questions will apply to each panel. However, your running through them quickly as you prepare to write an instruction keeps you aware that Apple Guide is dynamic and interactive and that, in essence, the user is at the other end of your keyboard.

What Information or Instruction Is Needed Here?

This question sets the scene. Its answer depends on what's gone before in the sequence. If an instruction has preceded this panel, must another instruction appear here? Or should the user be aware of some related information before taking the next action?

What Is the Computer's Status (from the Previous Step)?

You can assume the user has done the previous step (or read the information). Now what window is active? What application is in use? Is there a dialog box on the screen? Has the user just done a step that might have more than one specific outcome?

What Does the Computer's Status Have to Be for This Step?

Almost always in a sequence of actions, the current status should not differ from what it was after the previous step. If it does, then you may have skipped a step in the sequence or you may need to add to the previous instruction.

Author Note: This question and the preceding one are really helpful in combination. I've often discovered when comparing previous and current status from one panel to the next, that I assumed the status had changed but my instructions on the previous panel hadn't brought about that change. Invariably it's because I'd told the user to choose an option or name a document but had forgotten to add the phrase "then click OK" at the end of an instruction, so the dialog box I'd mentally dismissed was still on the screen.

What Else Must the User Know About This Step?
What Are the Consequences of This Action?

Sometimes a step has varying results, depending on the user's status before taking the step. For example, consider the instructions for checking a user's access privileges for a shared folder or disk. The user selects the item and then chooses Sharing from the File menu. The content of the window that appears depends on whether the user owns the computer where the item is located. Therefore the panel following the instructions to choose Sharing must explain both variations—the user doesn't own the computer where the item is located and the user does.

Networking and file sharing tasks are especially complicated, and most actions for a single panel won't require as much "if this, if that" explanation as the access privileges task. But the question of what else the user must know is a good reminder to examine a step from every possible angle so that you discover any variations as you plan and write your instructions, not after your Guide is finished.

What Background Information Might the User Need for This Step?

This question is especially useful for steps in which the user must choose one or more options in a control panel or dialog box. If the choices involve specialized vocabulary or features, you can't assume the user will know enough about each option to make an informed choice. In such cases, you can provide explanations, perhaps in a panel linked to the Huh? button.

In some cases, you may know that the user has to choose from among options, but you may not know what all the options are. This occurs, for example, with printing tasks because a user may have a version of a printer driver or a driver for a printer that the Guide doesn't recognize. You can provide users help with the options when you don't know the exact choices on the screen by having a panel advise them to turn on balloons to get information about the options.

What Might the User Do Wrong When Performing This Step (and How Can the Panel Text Anticipate the Problem)?

Because Apple Guide can't determine every detail of the status of a user's Macintosh, there will be some occasions when a Guide displays an instruction even if the user has not done the preceding step correctly. (This could be a fairly common situation if you don't use any context checks in a Guide.) Here you can assist users by anticipating such a situation and explaining what might be wrong if their screen doesn't match what the instructions describe.

For example, Apple Guide can detect that the user has an open Info window, but it can't read all the content of that window and so can't determine whether that window represents a program, a document, or a folder. So for a task such as changing the memory size for a program, you don't want to provide a coach mark at the location of the Memory Requirements section at the bottom of the window. (Technically, the coach mark can be drawn, but you can't determine whether the Info window that's open is for a program. If the Info window is for another kind of Finder object, the coach mark may appear over something else on the screen, completely outside the window.)

By anticipating such problems, you can "coach" users in the panel text. In our example here, you could advise them that if they don't see a section labeled "Memory Requirements" at the bottom of the

Info window, they haven't selected a program's icon. You can usually discover these instructional "gotchas" as you perform the steps of a task, particularly if you do the wrong thing and see what happens.

Does the User Need Any Special Navigation Instructions for This Step?

In most cases, the standard method you use for telling users how to move through a Guide is adequate. (The prompt line at the bottom of a panel usually provides navigation instructions. Guide Starter supplies a standard prompt that advises users to click the right or left arrow to move ahead or back.)

If a task requires users to do something unusual, such as clicking an OK button in two successive dialog boxes to complete a choice or clicking twice for an action that normally requires a single click, you may want to include a special instruction. For consistency, you should use the prompt line for any such navigation notes, unless you've chosen not to include prompts in your panels. (See "To Prompt or Not" in Chapter 7 for more about prompts.)

Are Any Tasks Closely Related to This Step? Are There Any Necessary Precursors or Logical Successors?

Sometimes a task that is a valid activity by itself also is closely connected to one or more other tasks. Sharing files on a network provides several examples of such tasks, including "How do I check my access privileges?" One panel in that task instructs users to connect to the shared disk that contains the item for which they want to check their access privileges. This instruction assumes that users know how to connect to a shared disk; the task would be unduly long if all the steps in the connecting task were integrated into this one. To assist users who don't know how to connect, however, the Huh? button on the panel leads to the instructions for how to do this.

You can use the Huh? button effectively to narrow the scope of a task and still offer users the backup information they may need. Using the Huh? button also enables you to point users toward a task that logically would accompany the current one. For example, a user who chooses the task "How do I change the labels in the Label menu?" in Macintosh Guide may also want to change the colors of labels, so a Huh? button would lead to that task.

For more information about using the Huh? button, see "Describe the Huh? Button's Content" in Chapter 7.

Is This Instruction Used for Other Tasks?

Many times a panel's instruction is appropriate for more than one task. If you determine that the panel you're writing can be reused, you may want to use less specific phrasing than you would for a panel that's dedicated to one task.

Can This Panel's Content Be Identical for All the Tasks It Is Used For?

When you've determined that a panel fits into more than one task, you may need to experiment with the wording in order to give it the widest use while still providing clear directions for users. For example, if your Guide has two or more tasks that tell users to copy a graphic, then a generic panel for those tasks could read "Open the graphics document you want to use." This instruction is specific enough, provided the panels before and after it refer to the purpose of the current task so that a user won't be confused.

As you plan instructions—and later as you go through your Guide—look for panels that could be used in several tasks if they are written generically. But also evaluate each generic panel within a task and write a variation for any instruction that isn't well served by the generic wording.

Use Basic Task Design with Guide Starter

Guide Starter automates the creation of a basic Guide database. A basic Guide database uses instructional text and coach marks as the primary elements to lead users through the steps of a task. In most cases, a basic database will meet users' instructional needs as well as your objectives.

Occasionally, however, you may want to add advanced features to the basic database, such as when the users of a Guide are novices (see the following note) or when a database requires very specific computer status to deliver the correct instructions.

Note: A basic Guide doesn't provide the kind of protected environment that's very desirable in a tutorial. Consider adding context checks and AppleScript modules (necessary for some coach marks and for auto-opening an item) in the text file if the users of your Guide will be computer novices.

You can always add advanced options to a Guide after creating the basic version. Your planning process should be the same in either case. That is, determine the relevant facts for each panel and make a note of them so that you have the information you need to later add a context check or an AppleScript component that opens an item for users. (See "Be Familiar with Advanced Features" later in this chapter for more information about advanced options.)

Concentrate on Instructions, Coach Marks, and Related Information

Using the basic design for Topics in a Guide involves doing the following:

- Writing all the instructions users need
- Providing a coach mark whenever possible to show users where to take the action described in a panel
- Ensuring you include information to assist users who get an unexpected result or who need to do another task before completing the current one

Take Advantage of Panel Features

The features built into Guide Starter's Panel Builder window give you the tools to create a basic Guide, and include the following:

- Coach marks to identify the location of an action
- Huh? buttons to provide links to definitions, background information, or related tasks
- Prompt lines to assist users as they navigate through the Guide

Decide which of these features to use as you plan each panel of a Topic.

The panel shown in Figure 6-1 provides a good example of the basic approach, in this case for the task "Where am I?" The example tells users how to know which program is the active one (panel 1, not illustrated) and how to switch to a different program (panel 2, shown in the figure).

The text on this panel delivers a simple instruction, which builds on the information in the preceding panel (that the active program has a check beside it in the Application menu). The second paragraph anticipates a question users might have when they open the Application menu: What is the Finder? Rather than adding a defini-

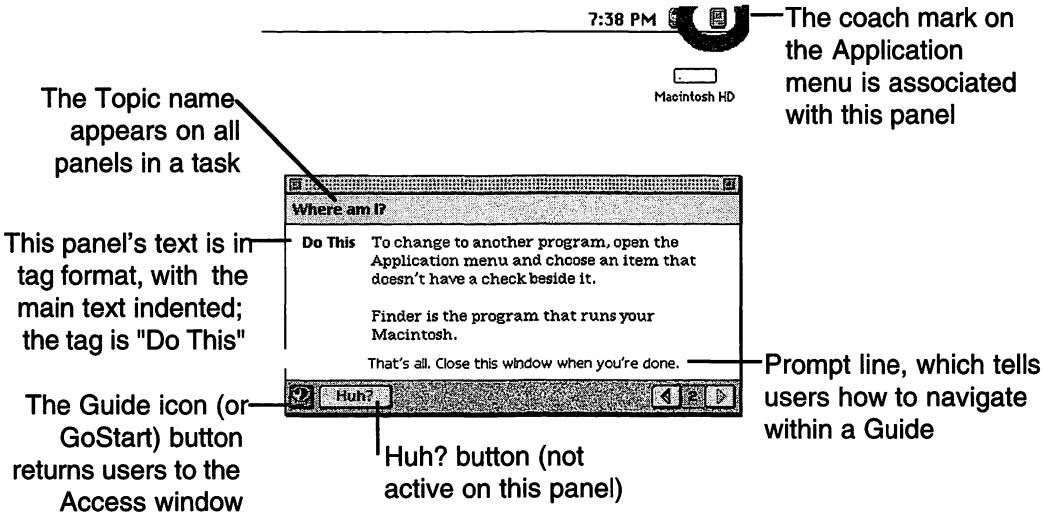


Figure 6-1
The primary elements of an instructional panel in basic task design

tion of the Finder that is accessed via the Huh? button, this panel's author took the straightforward course of providing this relevant fact on the panel itself. For users who need the help offered by this task, an up-front explanation of the Finder is likely to be welcome, while the panel has so little text that the extra sentence won't bog down or confuse readers.

The panel in Figure 6-1 has a coach mark to identify the Application menu so that users can quickly locate the site for their action. Because this is the last panel in the task, the prompt line advises users of that fact and suggests that they close the Guide window. The Huh? button isn't needed in this case because the necessary background information is in the panel text.

Provide Generic Help for Users' Mistakes

In the basic Guide database, coach marks show users where to take the action described. But if a user mistakenly changes the status of the computer before the coach mark appears (for example, by clicking outside the active window so that it becomes inactive), the coach mark is likely either to appear in the wrong place or to be muted because another window is on top of the location it is supposed to mark.

To assist users who may be confused by their own inadvertent actions or by a misplaced coach mark, you can provide some general troubleshooting tips. This is the sort of information the Huh? button is designed for, but you should also include a reference to the tips in the panel's main text. (In fact, it's a good idea always to tell users what content the Huh? button provides.)

The troubleshooting tips would consist of a series of brief explanations and suggestions like these:

- If the item circled on your screen doesn't seem to be what's described in the Guide's instructions, use the left arrow to move back to one or more previous steps and verify that you have taken the actions described in those steps. Then move forward again.
- As you follow the steps described in the Guide, be careful not to place the pointer on another area of the screen and click. If you are working in a window and you inadvertently click outside of it, that window is no longer active, and Apple Guide may "coach" (by circling or indicating with an arrow or an X) the wrong item or part of the screen.
- If you aren't sure you're using the right instructions or the right program, first check the name of the task at the top of Guide window. If the name doesn't seem familiar, then click the Guide icon button (at the lower left-hand corner) to return to the Topics list, where you can look for the task you want. To check which program you're using, open the Application menu at the upper right-hand corner of the screen. The item with a check beside it is the active program.
- You may get ahead of the instructions at times. If one of the Guide's instructions describes a step you've just completed, move to the next step. If its content is unfamiliar or you aren't certain about the instructions, you can easily go back and review the previous steps.
- If the Guide window covers an area where you need to take an action (the coach mark will be dim where the designated item is obscured), you can move the Guide window or the other window to see the hidden content. (You can also shrink the Guide window by clicking the box at its upper right-hand corner.) The coach mark disappears when you move either window. To display it again, go back to the previous step, then move again to the current one.

You will probably want to add other items that are specific to the subject of your Guide.

On some panels, you may want to use the Huh? button for different information even though the tips would be useful as well. If you have two sets of extra information, you could either include the most necessary excerpts in the panel's text and use the Huh? button for the rest or combine the two sets of information in the Huh? button content.

The amount of support and troubleshooting information you should include in a Guide depends on the users' level of computer experience and confidence. One practical approach is to link the set of tips to the Huh? button on the first two or three panels for which it might be needed and to discontinue using that information in later panels. Also, use the same set of tips or an expanded version as a Topic that's listed in the review and troubleshooting sections of the Guide's content so that users can easily find it if they have difficulty with a task.

Another way to handle troubleshooting is to create a "Why can't I" version of every "How do I" Topic. If doubling the number of action Topics isn't feasible (in a large database, for example), include with the product general troubleshooting information for each of the major activities.

Be Familiar with Advanced Features

As noted previously, you can add advanced features to a Guide by modifying the text file that you create with Guide Starter. We recommend that you first prepare a basic Guide and compile it and then try it out with a small number of users to determine if the instructions and level of support information are suitable for the intended audience. (See Chapter 10, "Testing and Revising Your Guide," for a detailed discussion of getting users' feedback to a Guide before its release.)

If you discover that users have difficulty following the instructions or that they are so unfamiliar with how to use the Macintosh that they become confused easily, consider building in context checks that establish a narrow, focused path through the instructions for each task. You may even want to have Apple Guide do some steps for users, such as opening a window or locating an item for them. You can use AppleScript components for these maneuvers.

Another advanced feature—branching tasks—can be very useful if your database is quite large or if it contains tasks with many steps. (Tasks that have more than twelve to fifteen steps can be daunting to anyone, particularly a user who's already a bit puzzled and perhaps frustrated as well.)

Build Branching Tasks

A branching task consolidates two or three similar tasks or presents several different features that can be used for the same purpose. For example, the Macintosh Guide task "How do I protect a file or disk?" includes three branches, one each for locking a file, locking a disk, and protecting two specific folders. Such combined tasks are efficient because they reduce the total number of tasks and often present the computer's uncommon vocabulary and concepts in terms that are familiar to users. (Table 5-1 presents additional examples of combined tasks.)

You plan and write the content for a branching task in the same way as for any other task, except you use modified scripting and an added "decision panel" to introduce the branches. As you plan your basic Guide, take note of tasks that might be combined as branches in a single task. Even though you write them as separate tasks initially, it's not difficult to combine them later by modifying the text file. Chapter 14 provides instructions for creating tasks with branches.

Use AppleScript to Locate or Open an Item

AppleScript is a powerful scripting language that can interact with the Finder and with many application programs. The Apple Guide technology recognizes AppleScript components, and the Guide databases included with System 7.5 use AppleScript to locate some items and to open others.

AppleScript's capability to locate items can be especially helpful if your Guide deals with documents or programs that users move around on their systems. Further, the Apple Guide technology sometimes isn't recognized by a program, in which case, you can often use an AppleScript module to find the appropriate area in the program and display a coach mark on it.

Chapter 16 explains the use of AppleScript components in a Guide and provides a set of examples for you to use and adapt to your needs.

Add Context Checks

Context checks monitor the computer's status and report it to Apple Guide. If your database has scripting to respond to whatever status the context check reports, you can greatly reduce or even eliminate the possibility that users will make an error or get lost as they follow your Guide's instructions.

For example, relatively simple scripting in a database can use a context check included in the core Apple Guide technology to make sure that the correct window (the one in which the user is to take an action) is open and active. If the check determines that the window is not open or active, then either the Guide can display a message telling the user to make the window active, or the database can open the window or make it active for the user.

Another simple context check can detect whether the user has moved ahead of the instructions, for example, by opening a control panel before the Guide's instructions have reached that step. The database can then skip the intervening panels and display the instruction that should appear after the control panel has been opened, thus presenting only the information needed.

As noted previously, context checking to monitor the user's actions and then respond with assistance is particularly important for a Tutorial or other Guide intended primarily for novices. If you are creating databases for novices or relatively inexperienced Macintosh users, consider providing context checks in your Guide. Chapter 17 explains how to include a number of standard context checks in a Guide's text file.

Respond to Users' Errors (Oops Panels)

If you include context checks in a Guide, you can write special panels, called Oops panels, in which you advise users of the incorrect status and direct them to the instruction that should remedy this situation. Or sometimes you may want the Guide to take the remedial action for the user. If so, you can write a variation of the Oops panel to tell users that the database is taking the action. In both cases, the special panel contains a button that the user must click before moving to the next panel. These response panels are discussed in Chapter 17.

Guide Ideas

A wide variety of instructional and reference Guides are well suited to the basic style of database. The examples that follow here and in the "Guide Ideas" section of other chapters may be helpful as you adapt the Apple Guide technology to the needs of your group or your clients.

Nifty Things You Can Do With [product]

Most people who use a computer for their work aren't especially curious about the computer or even about the advanced capabilities of the programs they use most often. You could introduce intermediate users to a program's fancy features and complex operations in a Guide that leads them through a set of advanced tasks. If the Guide's examples are crafted to be directly relevant to these users' work (or simply to their interests), your database could boost users' productivity and increase their sophistication with the Macintosh at the same time.
For everyone

Sending and Receiving a Fax

Many computers are equipped with fax modems, yet many users don't send or receive faxes often enough to memorize those two procedures. Your Guide could escort them through the two procedures (or you could create a separate database for each), providing hardware- and software-specific instructions and coach marks if there's some uniformity among their equipment. The instructions should also deal with telephone line connections and warn users that many modems are designed for use only with residential-type lines, not with PBX or other commercial lines. Even if the fax at your location is a separate device that isn't controlled by the Macintosh, many people would still use a Guide that reminds them of the procedures for using it. As with any database, the more specific the Guide's instructions, the more useful it will be for users.
For all fax users (even those who have stand-alone fax machines)

Teach Each Other

In any classroom or lab situation, the Macintosh is commonly used for structured projects and assignments. Often during work times, impatient experts crowd around the computer,

showing off their skills, while reluctant beginners remain at the fringes of the group. Apple Guide can be especially useful in these learning environments by providing a tutor on every machine. Better yet, the authors of Guides for the classroom and lab could be the local experts, who can apply their Mac savvy to creating databases for other students and teachers to use. Collaborative Guide-building could also bring mentoring to a class or group, with more experienced students sharing their skills and newcomers contributing Guide content by suggesting what tasks they need to learn. Student volunteers could create Guides for teaching staff (and vice versa), and parents and kids could give each other "assignments" for database projects.

For classrooms, computer labs, kids, teachers, and parents

SEVEN
CREATING THE PANELS

Once you've planned the panels for the Topics in your Guide, you can write the panel text and add panel options in Guide Starter. For simplicity's sake—and to emphasize some guidelines and tips for each part of the procedure—we've separated panel creation into two phases: writing text and adding panel options.

Of course there's no single or preferred way to enter the panel content. When you've had some practice writing Guide instructions, you'll probably want to complete all the content for one panel, move to the next panel and complete it, and so on. When you're using Guide Starter, you can always move forward and backward through the panels for a Topic or go to a different Topic to make changes in its content.

Hands On: Writing Panel Text and Adding Options

The objectives and guidelines that follow should be helpful as you write the instructions and explanations that users will read. Instructions for writing panel text appear at the end of this section.

Use Guide Starter's Panel Builder

You use the program's Panel Builder window, which opens when you choose a Topic in the Access Window Builder, to enter the panel text.

Follow these steps to begin writing panels for your Guide:

1. If necessary, make your Guide Starter application the active program, with the Access Window Builder the open window.

   If an Edit button is highlighted, click it to hide the editing extension.

2. Select a Topic Area in the list on the left side of the window.

   If your Guide is in Single-list format, skip this step.

3. Select a Topic in the list on the right side of the window.

   If you've selected a new task, a dialog box opens.

   If the task already has some content, then the Panel Builder window opens, and you can skip the next step.

4. In the dialog box, click New to open a new panel.

   The Panel Builder window appears (Figure 7-1), initially in its small form.

   (For information on assigning a panel, see "Assigning an Existing Panel to a Task" later in this chapter.)

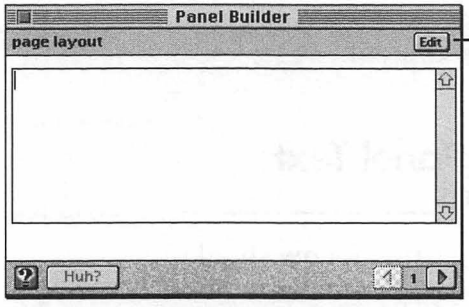


Figure 7-1
The Panel Builder window

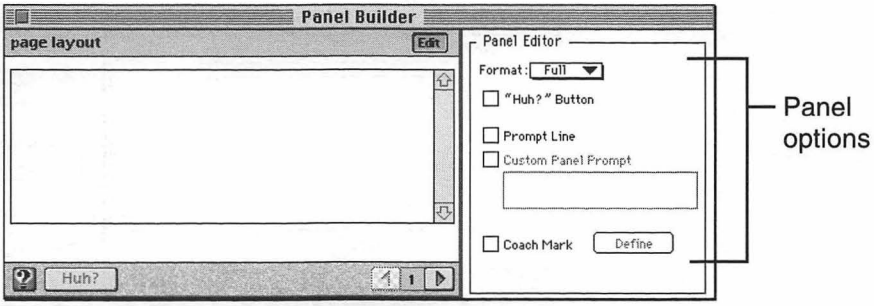


Figure 7-2
The expanded Panel Builder window

5. Click the Edit button to display the panel options section of the window (Figure 7-2).

   With the panel options displayed, you can change the panel's format and add a Huh? button, a prompt, and a coach mark. Once you expand the window, the options section stays open as you move from panel to panel.

Changing the Format for a New Panel

Before you write the first text on a panel, verify that the format is the one you want for that panel. The two formats available are Full (shown in Figure 7-4), which you use for introductions and explanatory text that doesn't include a direct instruction, and Tag (Figure 7-3), which you use for instructions.

The standard tag (at left, in bold type) for an instruction is "Do This." You can also use this format for other types of panels, such as tips, in which the tag is "Tip" (as used in Macintosh Guide, for example).

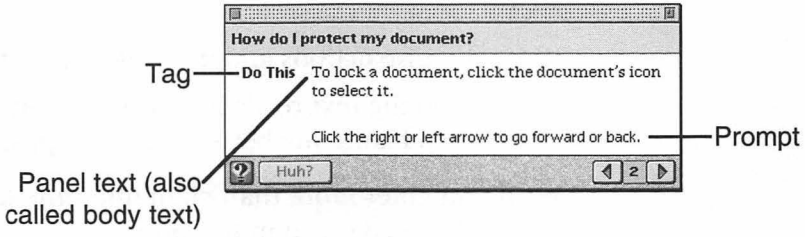


Figure 7-3
A panel in Tag format

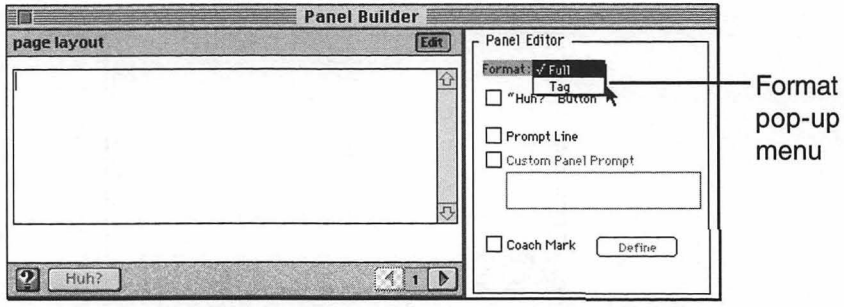


Figure 7-4
The Format pop-up menu in the Panel Builder window

To change the format for a panel, choose the format you want from the Format pop-up menu in the panel options section of the window (Figure 7-4).

Important: Be sure that a panel is in the format you want before you begin entering content. If you change the format after writing its text, you may lose the text you added in the original format. To avoid having to retype the text, you can copy it before changing formats, and then paste it in the new format.

Entering Text in a New Panel

After verifying that the panel format is right, you enter the text in the same way as for any standard word-oriented program.

✔ Follow these steps to write your instructions and information in a panel.

1. Click in the panel's text area (immediately below the task name) to set the insertion point, if necessary.

   Skip this step if the insertion point is already blinking in the text area.

2. Write the instructions and information for the panel.

   - ➤ To keep the text readable, use short sentences and paragraphs. Leave a blank line between paragraphs.
   - ➤ If you enter more than eight lines, the text area automatically scrolls to give you more room.

Important: The fonts used for panel text and for the Access window are specially designed for readability on the screen. The Starter Kit disk includes a copy of these fonts, Espy Serif and Espy Sans (regular and bold).

Apple Guide's design intentionally limits the font styles and sizes you can use in panels so that the on-screen instructions will have maximum readability and clarity.

➤ You can't change the font, size, or style of the text in the Panel Builder window. You can change the font used in the Presentation (panel) window and the Access window if you have a compelling reason to do so, but we recommend using the Espy fonts for all Guides.

➤ See "Entering Special Characters: Apostrophe, Quotation Marks, Symbols" in the next section for tips on using symbols and certain special characters in a panel.

3. Click the right arrow to move to the next panel.

   If no panel exists, you'll see the dialog box asking you to specify a new panel or to assign an existing panel as the next one.

   (The preceeding steps 1–3 cover entering text for a panel. Instructions for specifying other panel options are in "Add Panel Options" later in this chapter.)

Entering Special Characters: Apostrophe, Quotation Marks, Symbols

As you enter text in the Panel Builder window, you may want to use special characters such as the "curly" apostrophe and quotation marks or a bullet or other symbol. You use specific keyboard combinations for these characters.

Guide Starter enters the "straight" apostrophe or quotation mark in the Panel Builder window when you press the key that displays one of these characters. By using a key combination, however, you can enter the stylistically correct—and aesthetically preferable—curly versions of these characters. Table 7-1 shows the key combinations for the curly apostrophe and quotation marks. Note that it isn't

Table 7-1
Keyboard combinations for apostrophe, quotation marks, and attention-getting symbols

Character	Keyboard Combination
Apostrophe	Option-Shift-close bracket ( ])
Opening quotation mark	Option-open bracket ([ )
Closing quotation mark	Option-Shift-open bracket ([ )
Bullet (•)	Option-8 (in many fonts)
Diamond dingbat (❖)	Option-6 (in Espy only)
Triangle (△)	Option-j (in many fonts)

mandatory that you use the key combinations for quotation marks. Because the straight quotation mark is used extensively in the scripting for a database, Guide Starter changes straight quotes in panel text to the curly variety when it creates the text file for your Guide.

If you must highlight or separate items on a panel that contains a lot of text (especially if you can't leave blank lines between items), you may want to use a special character such as a bullet (•) or other symbol. Two good attention-getting symbols in the Espy Serif font used for panels are the diamond dingbat (❖) and the triangle, or delta (△). Keyboard combinations for these special characters are also shown in Table 7-1. You can see other symbols available in the Espy fonts with the Key Caps desk accessory in the Apple menu.

Adding Space Between the Text and the Prompt

The text on a panel and the panel's prompt serve different purposes. The text refers to something outside the Guide window; the prompt refers to the Guide itself. For this reason, these two groups of words should have enough space between them to make this distinction obvious.

When Guide Starter creates a text file and compiles a database, it puts at least one blank line between the text and the prompt (see Figure 7-3 for an example of the standard spacing). You can increase the standard space above the prompt by adding one or more carriage returns below the final text on the panel. Figure 7-5 shows panels with one, two, and three returns after the text.

Experiment with this spacing if you find that the standard space between the text and prompt is less than you prefer. Remember that if you add extra blank space on the panel, you'll be reducing the lines of text available on that panel by the number of blanks lines added. (A panel can have a maximum of fifteen lines, including blank ones.)

Whatever interval of space you use, be sure to use it consistently on all panels that contain prompts. Otherwise the text in the window will appear to jump around as users move from one panel to the next. This "movement" could be especially distracting because the Apple Guide window itself changes size for each panel (it adjusts to be just large enough for the text area so that it covers as little of the screen as possible).

Assigning an Existing Panel to a Topic

You can assign an existing panel to a topic at any point. Follow these steps to assign to the current Topic a panel that you wrote previously:

1. Click the right arrow of the last panel in a Topic.

   The dialog box for adding a panel appears.

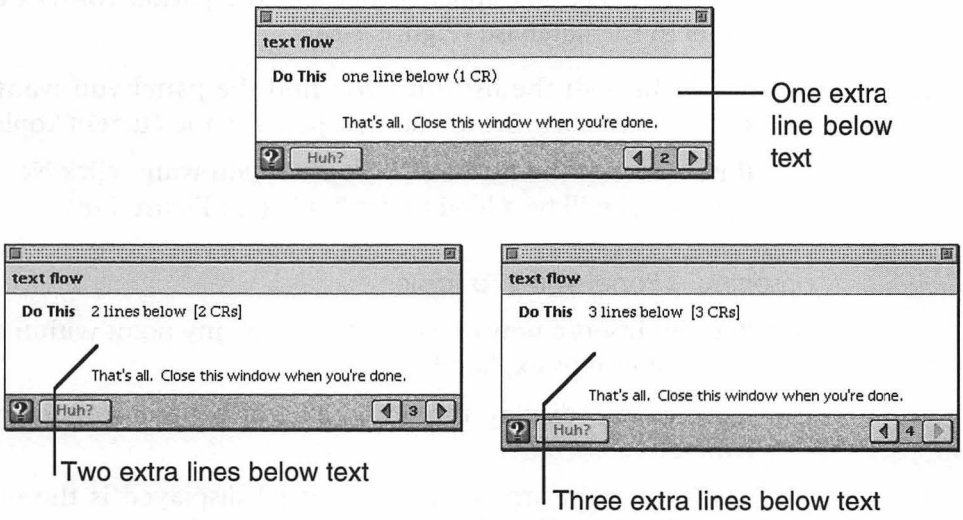


Figure 7-5
Three examples of extra space between text and prompt

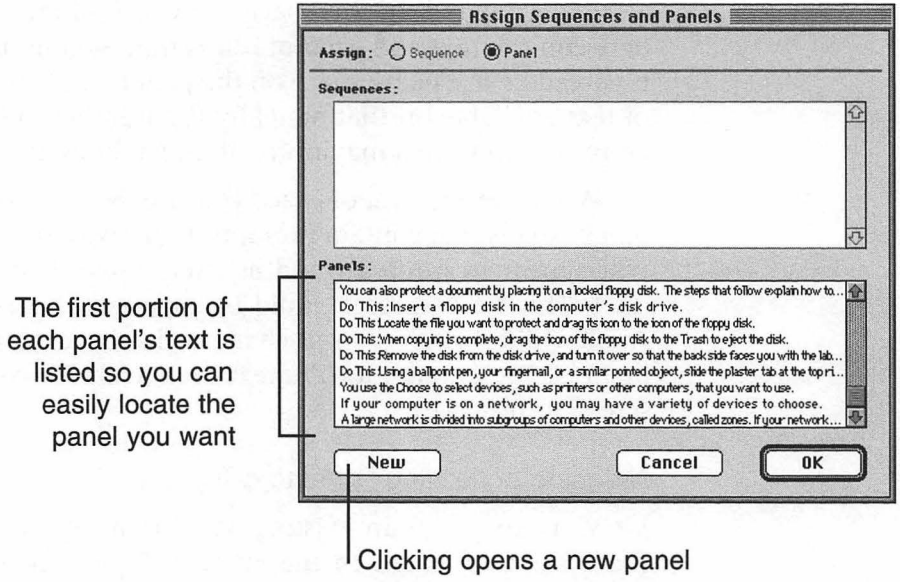


Figure 7-6
The Assign dialog box with all panels listed

2. Click the Assign button.

   A large dialog box appears that lists the partial content of each panel in the database (Figure 7-6).

3. Scroll through the list until you find the panel you want. Then select it and click OK to add that panel to the current Topic.

   If no panel in the list has the content you want, click New and a new panel will be added to the Topic (see Figure 7-8).

Inserting a Panel Within a Topic

You can insert a new or existing panel at any point within a Topic. The following steps explain how:

1. If necessary, display the Topic in which you want to insert a panel.
2. Click the right arrow until the panel displayed is the one that should appear after the panel to be inserted.
3. Choose Insert Panel Before from the Sequences menu.

4. Click the Assign button to insert an existing panel or the New button to insert a new panel.

   If you clicked Assign, you'll see the list of panels from which you can select a panel, as described in the preceding section.

Removing a Panel from a Topic

You can remove any panel from a Topic. If that panel has also been assigned to one or more other Topics, the panel will remain in those Topics.

Follow these steps to delete a panel:

1. Display the Topic from which you want to remove a panel.
2. Click the right arrow until the panel you want to remove is displayed.
3. Choose Remove This Panel from the Sequences menu.
4. A dialog box appears that asks you to confirm this action. Click OK to remove the panel. If you aren't sure you want to remove the panel, click Cancel.

You can go through the Topic to verify that you want to remove the panel and if so, repeat the steps above. As you consider whether to remove the panel from a Topic, consider whether the Topic (also called a sequence) appears in several places, such as linked to Huh? buttons or Index terms. Be sure that the panel will not be needed as part of the Topic at any other place that Topic appears.

When you remove a panel from a topic, you aren't deleting it from Guide Starter's data. It will appear in the list of all panels, and you can still assign it to topics in the usual way.

Add Panel Options

For the most part, you make the decision to use panel options as you design your Guide and as you go through the steps of each task in preparation to write instructions. As you write panel content, however, you may want to change your initial plan for some panels. And once you've had experience writing a database or two and seeing how users work with your Guides, you may be able to postpone deciding on panel options until you're writing the panels.

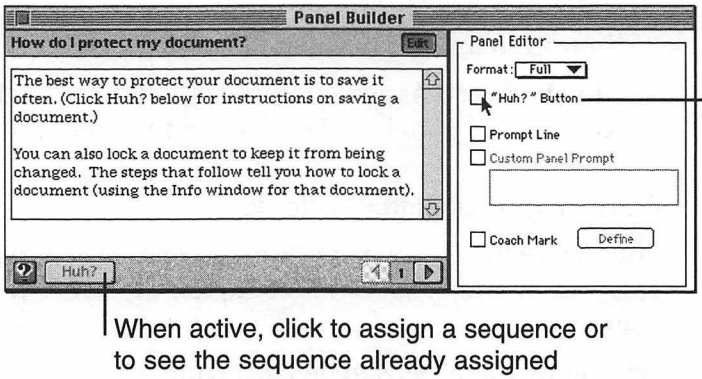


Figure 7-7
The Huh? Button checkbox activates the panel's Huh? button

Specify Huh? Button and Prompt in the Panel Builder Window

You can easily specify the Huh? button and prompt for each panel in the expanded Panel Builder window.

Assigning Content to the Huh? Button

Follow these steps to add content to the Huh? button:

1. Display the panel to which you want to add Huh? content and click the Edit button, if necessary, to display the options section (named Panel Editor).
2. Click to put an X in the box labeled "Huh? Button" (Figure 7-7).

   When you put an X in the checkbox, the Huh? button in the Panel Builder window becomes active, and a dialog box appears that asks if you want to assign a sequence to the Huh? button.
3. Click Assign in the dialog box.

   The Assign Sequences and Panels dialog box appears, with the list of sequences displayed (Figure 7-8).
4. Select a Sequence from the list and click OK.

   A Panel Builder window appears showing the sequence (Topic) you selected.
5. Verify that the Topic you selected is the correct one and then close its Panel Builder window.

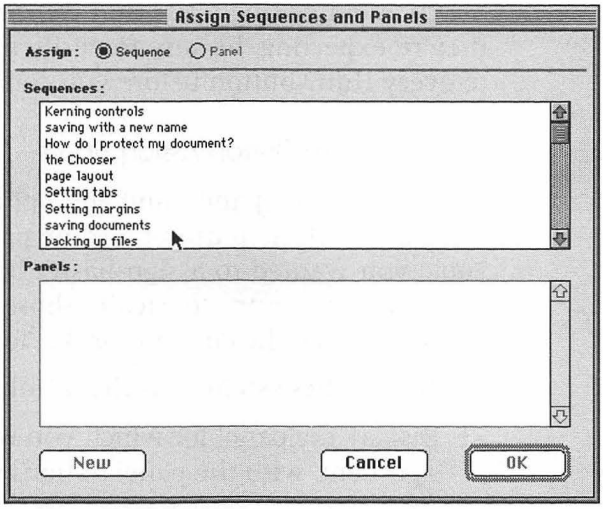


Figure 7-8
The Assign dialog box with a list of sequences for Huh? button assignment

If you find that you've chosen the wrong Topic, close the Panel Builder window on top. Then begin again by clicking the box labeled "Huh? Button" twice—first to remove the X and again to replace it. (After you click the first time, to remove the X, you'll see a dialog box asking you to confirm that you want to delete the assignment. Click OK.) Then repeat steps 3 through 5.

Important: In some cases you may need to activate the Huh? button on a panel without assigning a sequence to it. If you haven't created the panels for the Topic you want to assign to that Huh? button, you'll have to click Cancel in the Assign dialog box and return to the Topic later to make the assignment.

If you activate the Huh? button on a panel (by putting an X in the box labeled "Huh? Button") but don't assign a sequence to it, your Guide will display Guide Starter's placeholder panel (Figure 7-9) when users click Huh? on that panel.

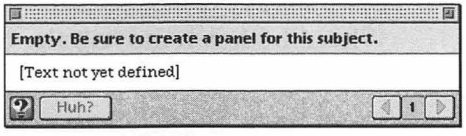


Figure 7-9
Guide Starter's placeholder panel

To avoid having users find this panel instead of the information they're expecting, be sure to verify that you've assigned a sequence to every Huh? button before you compile your Guide.

Verifying Huh? Button Assignments

When you write panels and add options to them, you probably will activate the Huh? button on some panels and then discover that the Topic you wanted to assign hasn't been created yet. In these cases, you'll have to return to each of those panels after you're certain that you've entered the content for the Topics to be assigned.

Follow these steps to verify a Huh? button assignment:

1. Display the panel for which you want to verify a Huh? button assignment, with the panel options section open.

   If necessary, use the Edit button to open the options section (Panel Editor).

2. Click the Huh? button at the bottom of the Panel Builder window.

   Either the Panel Builder window opens for the Topic assigned to the Huh? button, or, if no assignment has been made, a dialog box appears asking if you want to make an assignment.

3. Verify that the Topic displayed is the one you want. Or assign a Topic by following the procedure in "Assigning Content to the Huh? Button" earlier in this chapter.

Removing Content from the Huh? Button

Follow these steps to remove a Huh? button assignment:

1. Display the panel from which you want to remove a Huh? button assignment, with the panel options section open.

   If necessary, use the Edit button to open the options section (Panel Editor).

2. Click the box labeled "Huh? Button" to remove the X. Then click OK in the dialog box that appears.

   When the X is removed, the Huh? button becomes inactive.

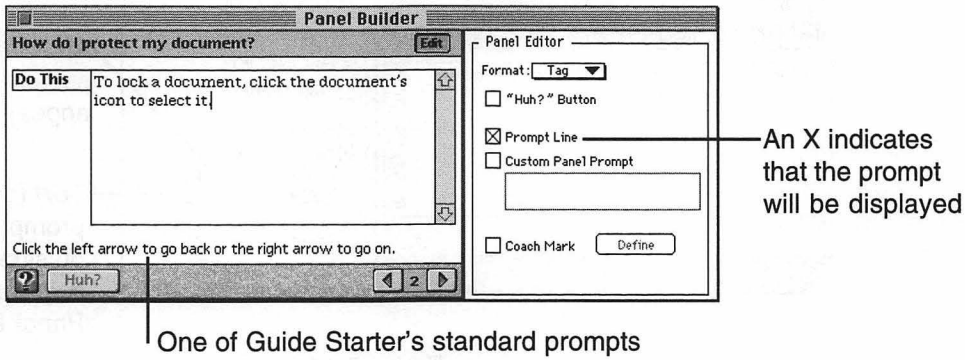


Figure 7-10
The Panel Builder window with a standard prompt displayed

Using a Prompt—Built-in or Custom

Follow these steps to display a prompt line on a panel:

1. Display the panel on which you want to use the prompt, with the panel options section open.

   If necessary, use the Edit button to open the options section (Panel Editor).

2. Click to put an X in the box labeled "Prompt Line."

   One of Guide Starter's standard prompts appears at the bottom of the Panel Builder window when you put the X in the box (Figure 7-10).

   (The standard prompt may change when you click the right arrow to move to the next panel because the program's standard prompts differ depending on a panel's location in the sequence.)

   If you want to use the standard prompt, you're done.

3. To write a custom prompt, click the box labeled "Custom Panel Prompt" to put an X in it.

   The standard prompt line disappears from the panel when you click the custom prompt box.

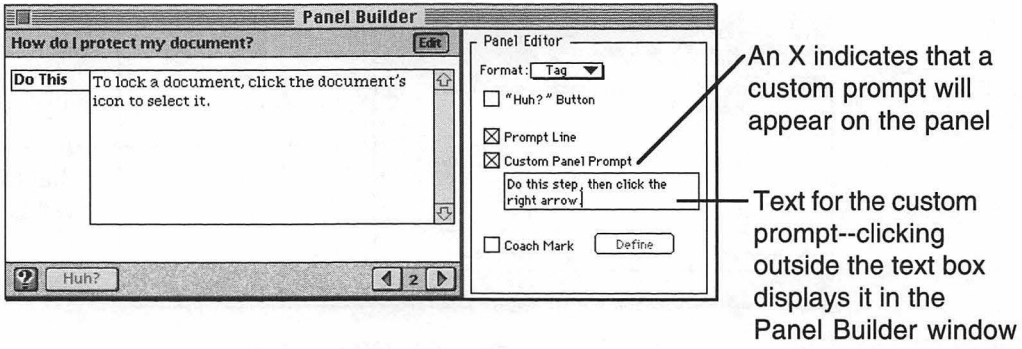


Figure 7-11
The text for a custom prompt

4. Click in the text box below Custom Panel Prompt to set an insertion point. Then type the text for the prompt (Figure 7-11).

   The text you typed appears as the prompt line when you click anywhere else in the Panel Builder window or press the Enter key.

Keep your custom prompts as brief and clear as possible so that users can read them quickly and follow their directions easily.

Specify Coach Marks in the Panel Builder Window

You can add any of several coach marks with Guide Starter by using the options section of the Panel Builder window. For one type of coach mark, you must provide some information about menu choices; for another type, you need to identify the corners of the area for the circle.

Adding a Coach Mark on a Menu

Follow these steps to add a coach mark to a menu title and an item in that menu:

1. Display the panel on which you want to add a coach mark, with the panel options section open.

   If necessary, use the Edit button to open the options section (Panel Editor).

2. Click to put an X in the box labeled "Coach Mark" (see Figure 7-12).

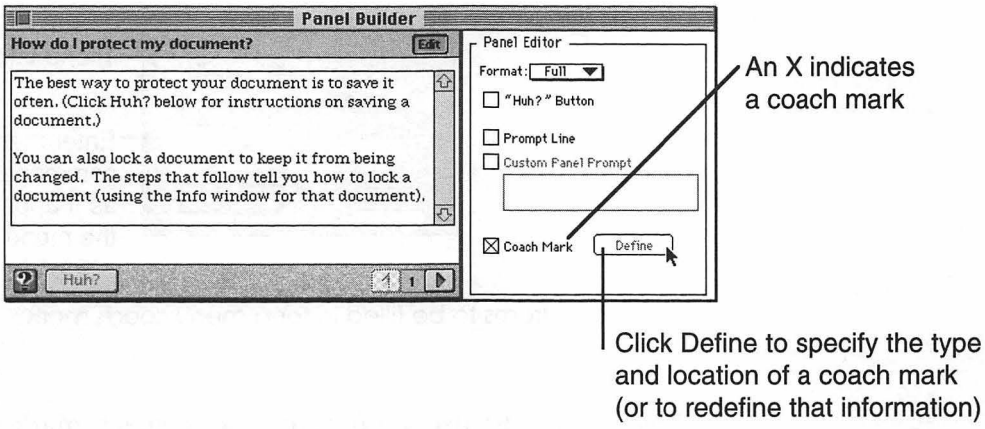


Figure 7-12
Coach mark activation in the Panel Builder window

3. Click Define to specify the coach mark type and location (Figure 7-12).

   The Coach Mark Builder dialog box appears with a pop-up menu of coach mark styles. The style used most recently is showing; if no coach marks have been specified, the menu shows None.

4. Open the Coach Mark Style pop-up menu and choose Menu (Figure 7-13).

   The dialog box changes to show the menu style.

5. Type the name of the menu you want to coach (Figure 7-14).

   Usually the title in the menu bar is the menu's name.

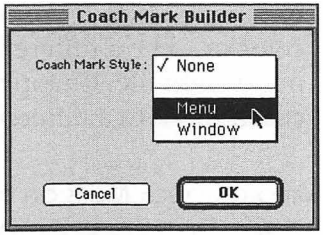


Figure 7-13
Pop-up menu of coach mark styles

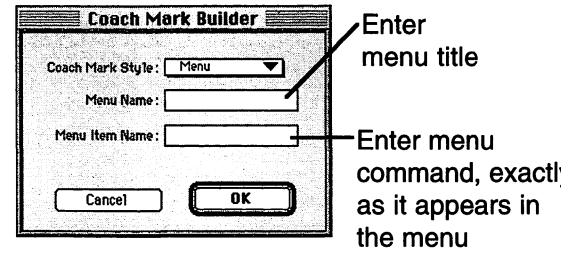


Figure 7-14
Items to be filled in for a menu coach mark

Note: You must use a different type of coach mark to coach the Guide (Help) menu or the Application menu. See Chapter 12 for more information.

6. Type the name of the item in the menu you want to coach (see Figure 7-14).

   Important: The text you enter must match the menu item exactly, including punctuation (but excluding the keyboard shortcut at the far right, if one appears). If the command is followed by an ellipsis, such as "Print..." in the File menu, use the keyboard combination Option-semicolon for the ellipsis, not three periods. (If a menu command containing an ellipsis is not coached after you compile your Guide, try three periods as an alternative. But start with Option-semicolon because Apple's guidelines specify that programs should use that key combination for an ellipsis.)

7. Click OK.

   Guide Starter adds the standard Apple Guide menu coach mark: a red half-circle on the menu title and red text and underline on the menu item. If you want a menu coach mark to be a different color or to be a different type style, you can change its definition on the text file. Chapter 12 provides instructions for changing the coach mark to a different style or color.

Adding a Coach Mark on a Window Element

✓ Follow these steps to add a coach mark to one of the standard elements of a window's border: the title bar, close box, zoom box, or size box (called "growbox" by Apple Guide):

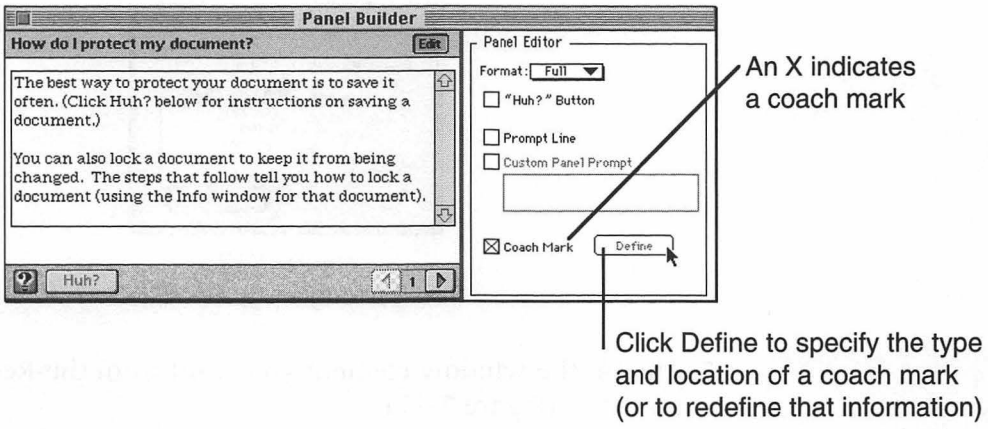


Figure 7-15
Coach mark activation in the Panel Builder window

1. Display the panel on which you want to add a coach mark, with the panel options section open.

   If necessary, use the Edit button to open the options section (Panel Editor).

2. Click to put an X in the box labeled "Coach Mark" (see Figure 7-15).
3. Click Define to specify the coach mark type and location (Figure 7-15).
4. Open the Coach Mark Style pop-up menu and choose Window (Figure 7-16).

Figure 7-16
The Coach Mark Style pop-up menu

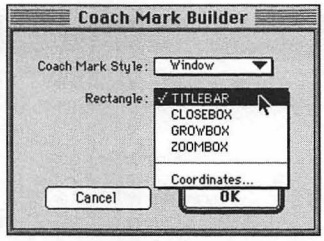


Figure 7-17
The Rectangle pop-up menu

5. Choose the window element you want from the Rectangle pop-up menu (Figure 7-17).

   The standard elements that Guide Starter coaches—title bar, close box, size box, and zoom box—are listed in the top portion of the menu. Figure 7-18 shows a coach mark on each of the window elements.

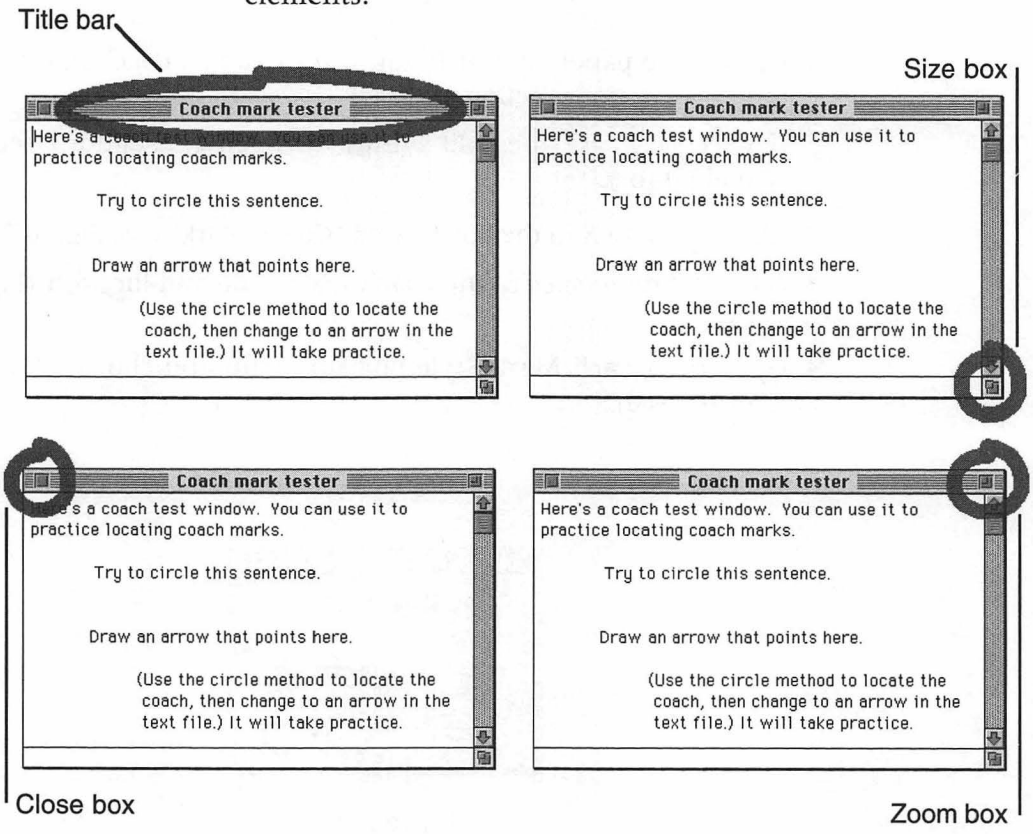


Figure 7-18
Coach marks on each of the four window elements

(Instructions for using the Coordinates item are in the next section, "Adding a Coach Mark on an Item Inside a Window.")

6. Click OK.

   Guide Starter adds the standard Apple Guide coach mark—a red circle—on the window element you indicated.

Adding a Coach Mark on an Item Inside a Window

Follow these steps to add a coach mark on an item inside a window:

1. Display the panel on which you want to add a coach mark, with the panel options section open.

   If necessary, use the Edit button to open the options section (Panel Editor).

2. Click to put an X in the box labeled "Coach Mark" (see Figure 7-19).
3. Click Define to specify the coach mark type and location (Figure 7-19).

   The Coach Mark Builder dialog box appears.

4. Open the Coach Mark Style pop-up menu and choose Window (Figure 7-20).

   The Rectangle pop-up menu appears in the dialog box.

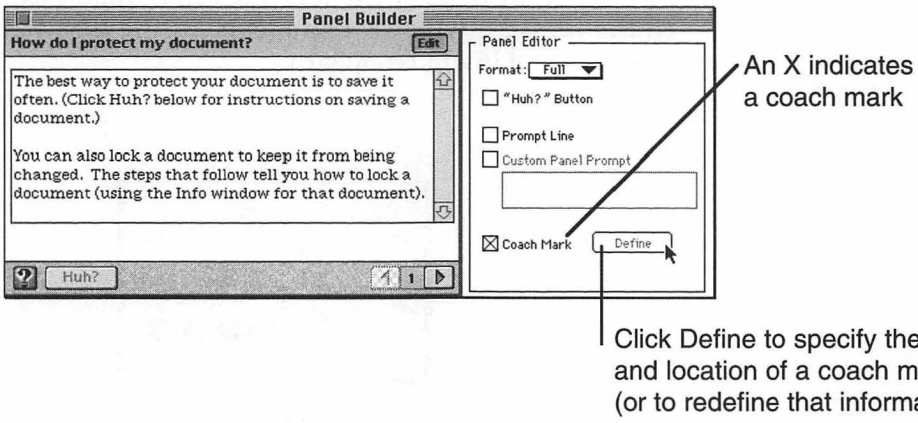


Figure 7-19
Coach mark activation in the Panel Builder window

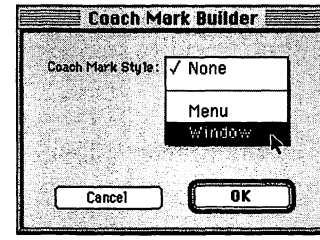


Figure 7-20
The Coach Mark Style pop-up menu

5. Choose Coordinates from the Rectangle pop-up menu (Figure 7-21).

   More content is added to Coach Mark Builder.

6. Click the New button to set coordinates for the coach mark (Figure 7-22).

   The Guide Starter windows close, and a tiny window appears at the lower right-hand corner of the screen. The pointer changes to small intersecting lines with an arrow pointing to their junction (see Figure 7-23). If you associated an application program with your database in the Preferences, Guide Starter makes that program active.

7. Place the "corner" pointer at the upper left-hand corner of the content area (immediately inside the border) of the window where you want the coach mark and click (Figure 7-23).

   The pointer changes to a slightly wider corner marker, and a new message appears in the box at the lower right (see Figure 7-24). (If the message window disappears during this transition, wait a moment and it will reappear.)

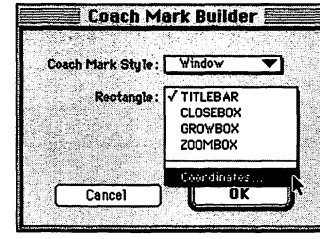


Figure 7-21
The Coordinates command lets you set a location for a coach mark

Figure 7-22
The New button initiates setting coordinates for a coach mark

8. Click at the upper left-hand corner of the area you want the coach mark to cover and click (Figure 7-24).

   Again the pointer changes after you click, this time to indicate the lower right-hand corner. The message changes as well (see Figure 7-25).

9. Click at the lower right-hand corner of the area you want the coach mark to cover (Figure 7-25).

   The message window closes and Guide Starter's windows reappear. Four numbers appear in the checkboxes labeled "Top,Left" and "Bottom,Right." These numbers represent the locations in the window where Apple Guide will draw the coach mark.

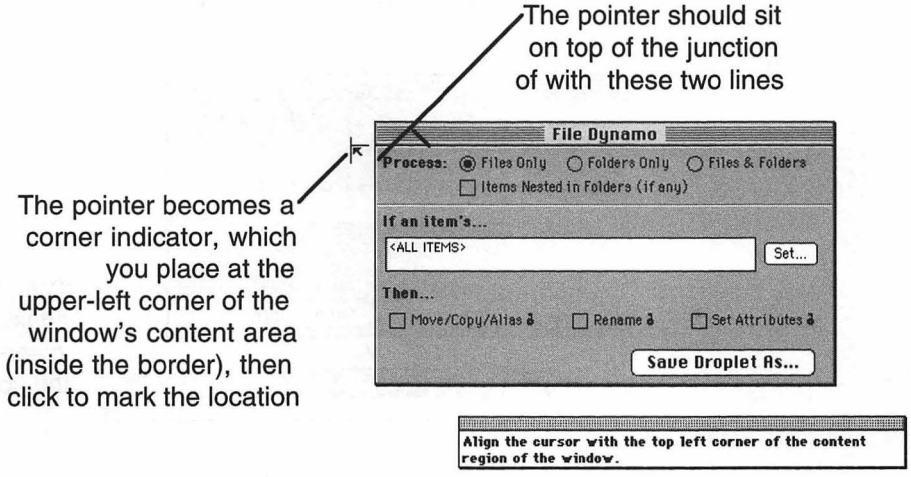


Figure 7-23
The corner marker for a window that will get a coach mark

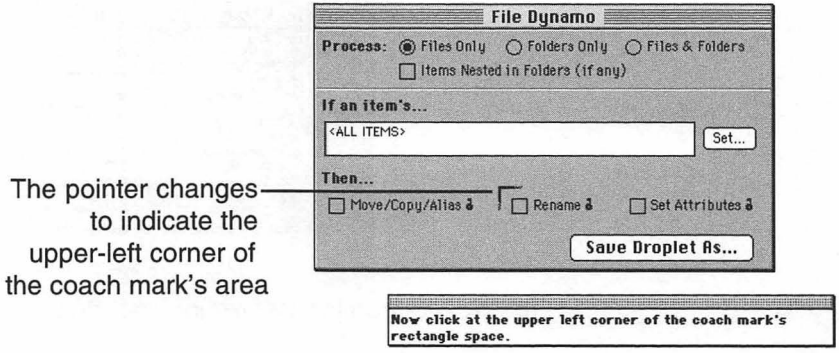


Figure 7-24
The marker for the top left-hand corner of a coach mark

10. Click OK.

    If necessary, you can adjust the settings (now or later) by changing any of the numbers or marking new locations.

Revising or Removing a Coach Mark

To change a coach mark, click the Define button and indicate the style. Then either type the new information (for a menu coach) or choose a new item from the pop-up menu (for a window coach). If you're setting new coordinates for a window coach mark, you'll need to indicate their locations as well.

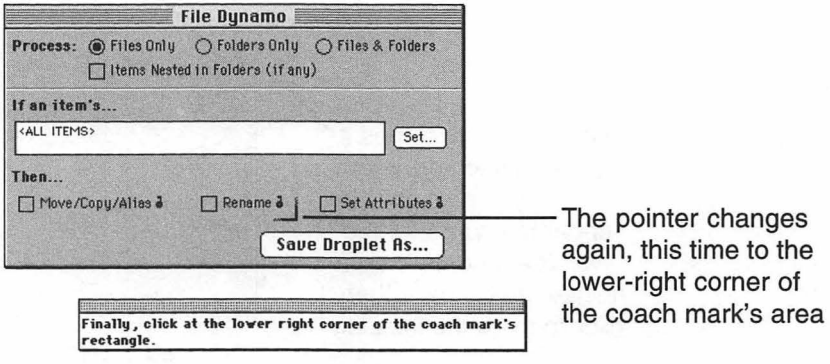


Figure 7-25
The marker for the bottom right-hand corner of a coach mark

To remove a coach mark, click to delete the X from the coach mark box. If you later want to use the coach mark after all, click the checkbox and then click the Define button. The Guide Starter program remembers the coach mark you specified originally and will display it in the Coach Mark Builder window.

Using Other Types of Coach Marks

Guide Starter uses the standard styles of coach marks: red circles and red, underlined menu items. You can modify the color and style of coach marks in the text file for your Guide (see Chapter 12).

You also can use advanced scripting and AppleScript modules to produce coach marks on items that require special treatment, such as dialog boxes or files that users are likely to move around. Examples of these specialized instructions for coach marks are in Chapters 12 and 16.

Head Work: Panel-Writing Guidelines

The guidelines that follow will help you write effective instructions for a database. They provide useful background and examples you can adapt to the needs of your Guide's users.

Write for the Demands of Online Instruction

As explained in Chapter 4, online instruction presents some limitations and demands that you can meet with careful planning and writing in your Guide.

Be Consistent Throughout

Because users develop a set of expectations quickly when using a Guide, your tasks and instructions should be consistent in all parts of the database. For example, if two tasks have similar content, avoid using three panels for one and eight for the other. Likewise, when writing procedural steps, use the same wording for comparable actions. For example, if you use "Open the File menu and choose Print" for one task, don't use "Pull down the File menu and select the Print command" elsewhere.

Keep It Short

Closely related to consistent design and wording is the genuine need for brevity, since users won't read much text on the screen. Use only the information users require for the action or explanation on a given panel. Attach variations and asides (if used at all) to the Huh? button or add them near the end of the task as tips.

The standard Apple Guide panel can display up to fifteen lines of text (including blank lines between paragraphs), but you'll rapidly lose readers if many of your instructions are that long. Panels with five to nine lines of text are usually a good compromise between brevity and impatience.

The length of tasks also should be limited. A practical maximum number of panels for a task is fourteen or fifteen, although some complicated operations may require more steps (and thus more panels). If a task is pushing the dozen-panel barrier, look at its content to see if any subtasks could be pulled out and linked to the Huh? button on an appropriate panel.

Another way to keep the instructions for a task short and still provide adequate background information is to create some Topics under an "About" heading or similar informational grouping. The About items could orient users who are unfamiliar with the subject so that they won't need explanations as part of the related tasks' steps. Figure 7-26 shows the Access window for a Guide with several About items.

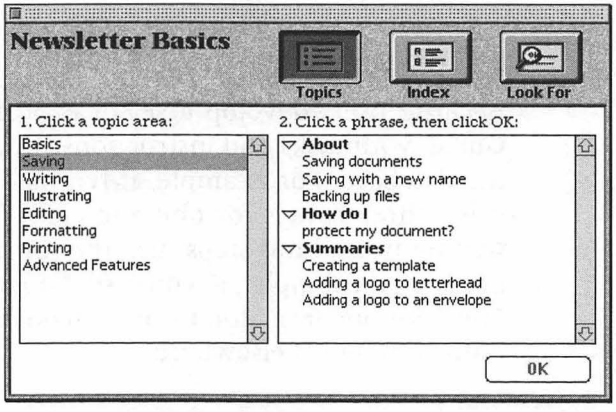


Figure 7-26
Access window showing Topics for "About" and "Summaries"

In some cases, you may want to give users more extensive background information or explanation than is feasible in a series of Guide panels. One alternative that might serve the needs of a work group is to use an AppleScript component to open a separate document that users could scroll through on the screen and even print. This document could be delivered with one of the many document-viewing utilities, which allow users to open and use a document without having the program that created it. (See Chapter 16 for more about using AppleScript with a Guide.)

If the users of your Guide are experienced and comfortable with the Macintosh, another strategy for brevity is to have the main type of instructions be summaries of task steps rather than step-by-step, coach-intensive help. If you're considering using summaries extensively, it's a good idea to consult with representative users in advance to make sure this approach meets their needs. Figure 7-27 shows an example of a one-panel summary of the steps for creating a template.

Summaries like the one in Figure 7-27 could also be presented as one branch of a task, with users having a choice of step-by-step instructions on multiple panels or a short list of steps on one panel. Chapter 14 includes an explanation of modifying the text file for a Guide to include these options as branches in a task.

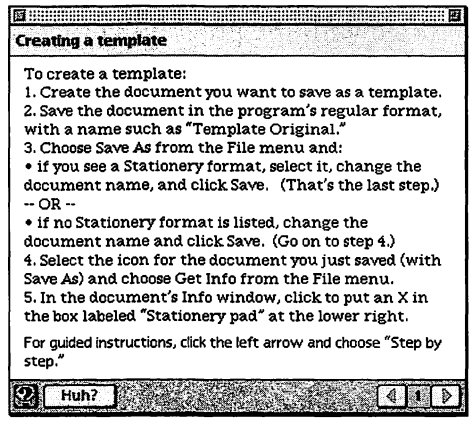


Figure 7-27
A summary panel that lists all steps for creating a template

Tell Users What's Coming

Because users can't see how extensive a task is, you can use the introductory panel to give them an idea of what's involved. Your introductory text should explain what the task is (and possibly why users would want to do it) and summarize the key actions. This method also enables confident users to try the task without going through the instructions.

Here's an example of text for the first panel in the step-by-step instructions for the task "How do I create a template?":

> *A template is a master document that you design and save in a format named Stationery. Whenever you open a template, a copy of it opens and you must give the copy a name that's different from the template name.*
>
> *You create a template in one of two ways: by saving it as Stationery (if your program provides that option) or by opening the document's Info window and putting an X in the box labeled 'Stationery pad.' The steps that follow explain how.*

In addition, consider including cross-references to other tasks and information. There's a good possibility that users will go through the instructions for a task and then realize at the end that it's not quite what they wanted to do. This is particularly true if your Guide covers software, such as a spreadsheet, that has many tasks with similar subject matter. A brief description of related tasks could really be useful in helping users find the task they want or just in acquainting them with the program's other capabilities.

Depending on the length of tasks and your use of the Huh? button for other information, you could attach a short list of related tasks to the Huh? button or use that button to open a single related task. For a Guide with much related content (as a spreadsheet's is likely to have), you may want to establish a standard format for providing cross-references at the end of each task. In this way, users can move ahead to that list if they discover that the instructions they're using are not the right ones.

Similarly, you can provide tips for specific tasks or for using tools or commands in a program at the end of a task or at a transition point in the instructions. Ideally, a tip shouldn't interrupt the sequence of actions. However, in some cases it may have to appear on a panel (or in the Huh? button) for users to take advantage of the information.

Don't Assume Everything Is OK

In addition to giving users a summary at the beginning of a task, you will find it a good idea to anticipate any problems they may encounter when following the steps. You can use the Huh? button to include troubleshooting tips like those presented in Chapter 6 (in the section "Provide Generic Help for Users' Mistakes") or provide brief, targeted recovery information on panels that you think may cause difficulty for some users.

Many tasks present enough potential hazards that you might also want to create a troubleshooting version of the instructions—the "Why can't I" Topic for that procedure. This type of task should contain the tips you provided for individual panels, plus more general suggestions such as making sure the cables are connected, the hardware configuration is appropriate, and so on.

Write for Efficiency—Yours and the Users'

In general, the more streamlined you can make your Guide's instructions, the leaner your database will be and the faster its users will get on with their work. Ideally, you want your instructions to keep users moving ahead when they're doing a task and to give them quick and easy access to the instructions they need when they're looking for a task.

One Panel, One Action

Keep your instructions consistent and avoid causing users to hesitate or become confused by ensuring that each panel contains only one action. Often the interface will guide you in this effort, because one action tends to cause a change on the screen, the next action causes another change, and so on.

Note: A panel can have only one coach mark associated with it. This is another reason for limiting a panel's content to a single action.

Of course you can use some flexibility with panel content, depending on the experience level of the Guide's users and the complexity of the tasks in the database. For example, you may cover two or three actions that users take in a dialog box in one panel. Or you could advise users to choose from among the options in a control panel or dialog box, rather than devoting a panel to each possible choice.

As you build a body of completed panels and tasks, you'll get a sense of the right balance for the subject and the users of your database. Don't be reluctant to adjust your original definition of "one action" if you find that the content fits a different model or that the users can readily follow instructions with more than one action per panel.

Think Generically

Your database will be more compact and its users will always see an instruction presented the same way if you reuse panels whenever possible. This strategy requires you to think about the "intersections" between tasks and use wording for the intersection panels that doesn't refer to a specific purpose, such as locking a file or changing a program's requirements. Instead, write a direct statement that tells users what to do where, such as "Click the item you want to select it" or "Place the CD in the tray and push the tray inward to close it."

Considerations for Panel Options

When adding panel options, keep the following guidelines in mind.

Provide Further Assistance

As noted previously, the Huh? button can clear up the mystery for a baffled user by supplying a crucial definition or explanation. It also can provide the steps for a task that must precede the current one. Both of these purposes are valid uses for this supplementary window, but a Guide can have too much of a good thing. If users see that the Huh? button is active on nearly every panel, this feature will lose some of its immediacy.

Because the Huh? button opens a second Presentation window, it has the potential to confuse some users or to lead them away from their original task. Take care to avoid linking long or complex sequences to the Huh? button. Doing this will minimize the chances of misleading users.

Also, try to avoid using panels that have an active Huh? button on them as part of content that you assign to a Huh? button. If the Huh? button opens a panel with a Huh? button, users could speedily get lost in an endless Huh? loop—the on-screen equivalent of those paintings that show a person looking at a painting of a person looking at a painting ...

For all the foregoing reasons, we suggest you try to limit the use of the Huh? button to essential information or closely related tasks.

Describe the Huh? Button's Content

Whenever you use the Huh? button, it's a good idea to state in the panel text what information the Huh? button contains. As with all elements in a Guide, you want to make the users' learning path (or task completion) as smooth and simple as possible. If you aren't sure that the content of the Huh? button will be obvious to users, explain it in the text.

To Prompt or Not

The prompt is designed to help users navigate through a Guide. Depending on the subject of your database and the experience of its users, you may want to omit prompts from most or all panels. If you write a custom prompt for only a few panels that need special treatment, users will certainly notice those messages.

If you aren't certain whether to include prompts routinely in a database, conduct a short test. Write some panels with prompts and some others without. Then compile a small database and try out the instructions, preferably with users whose experience matches that of the audience for your database. If your Guide's users are familiar with Apple Guide or use similar on-screen navigation, they may not need the extra line of text. On the other hand, if you plan to use lots of tips, explanations, and other information among the action panels for tasks, users will probably find the prompts helpful.

We recommend that you use a prompt on any panel that shows a graphic representation of the interface. Because users frequently mistake a picture of an interface element for the real thing, include a prompt that says "Example only" or a similar notation to alert users to the picture. (In general, you should be able to avoid using pictures because the coach mark will guide users to the desired interface element.)

Important: When you do use prompts, use them consistently. For example, don't include them on some panels and omit them on others in a task (unless you're using a prompt only on panels with very unusual content). Maintain the same style and wording for each type of prompt—action panel, tip, definition, and so on—throughout the database. If you need inspiration, then (as we've suggested before) look at other Guides for ideas and examples.

Coach Early, Coach Often

Coach marks are the heart of Apple Guide's design. You should use them whenever possible to show users where to take an action. Apple Guide's (and Guide Starter's) preset coach mark is a red circle, although you can specify either an arrow or an X instead. (You can use the arrow or X style of coach mark by adding it to the text file for your Guide, as explained in Chapter 12.)

When you designate coordinates for a coach mark using Guide Starter, keep in mind that a coach mark that covers a large area on the screen takes a relatively long time to draw, particularly on an older-model Macintosh. Always check the finished coach mark when you've assigned coordinates for its area; you may discover that it obscures part of the content that users need to see.

If your Guide will be used frequently on PowerBooks, you *must* check all the coach marks to ensure they don't cover necessary parts of the content. Try using the X instead of a circle for PowerBook instructions if the standard circle covers too much of the screen. When you use a variation of coach marking, be sure to apply it consistently in the database.

Occasionally you may find that a coach mark doesn't work with the software for which you're writing instructions. If this occurs, you may be able to use an AppleScript component to add a coach mark to parts of that software (see Chapter 16 for more information about AppleScript).

If you can't get a coach mark to work and it's important to show users the interface an instruction describes, include a picture of that part of the interface. However, because users often think the picture is the actual interface, use a picture only when absolutely necessary. When you do so, consider adding it to the Huh? button (with a reference in the panel text) so that it's available but not a prominent part of the instructions. Chapter 13 explains how to add a picture to a panel.

Another alternative for some situations in which you can't get a coach mark to work is to advise users to turn on balloons. For example, if your Guide instructs users to choose options in the Print dialog box, you must know which printer and which version of the printer driver are being used in order to ensure that a coach mark will appear in the right place. In such cases, your instructions could remind users that balloons provide descriptions of the options.

Don't Coach (Sometimes)

In some instances, not adding a coach mark is preferable to using one. For example, if the program your Guide covers has windows that change size according to the user's tasks, try to avoid assigning coach marks to panels that cover actions in which the window could be either of two sizes. One such program is AppleCD Audio Player, provided with System 7.5, which plays audio compact discs. Its window is compact when the user is simply playing a CD, but the window expands when the user is programming a playlist. So if you are writing instructions for making a playlist, refrain from using coach marks unless you can determine that the window is expanded. (See Chapter 17 for information about using context checks to detect a program's status.)

Keep Notes

As you write panels and add options, be sure to keep a record of content that needs further attention. This is also a good time to start—or add to—lists of terms related to the content of each Topic.

Your notes can take any form, but you'll probably want to arrange them for easy searching later. You can use your word processing program for this file. Consider organizing the notes by task, with a separate item for each panel. You may want to set up a template for tracking information about panels as you develop and revise a Guide database. If you used a template, you could adapt it for planning panels (see Table 6-1 in Chapter 6 for an example).

If your computer doesn't have enough memory to have the word processing program open while you're using Guide Starter, use the SimpleText program or Stickies (in the Apple menu) to record the notes and then open them in the word processor when you want to use its search features. (If you use Stickies for notes, you'll need to export the text to a word processing file for searching.)

Unfinished Items

As you create tasks and Topics, keep notes of any content that you need to verify or complete later. For example, if you activated Huh? buttons on some panels but did not assign content to them, note each one by the Topic in which it appears and its location within that Topic.

When you're revising the content of a database, it's important to keep a record of every change you make. Your change history should include each Topic that has been edited, the panels in the Topic that you revised, and the types of changes you made, such as text changes, prompt changes, and options added, removed, or changed.

Index Terms and Synonyms

Because you focus closely on the subject of your Guide as you write the panels, when you are writing them is a good time to record the key terms and phrases that relate to each topic. You'll use many of these terms in the Index (see Chapter 8), and you may add others to a list of synonyms if you decide to implement the Look For component.

(Look For allows users to type a word or phrase that Apple Guide evaluates. If it matches an Index item or any of the synonyms in a list you provide, the Topics associated with the matched synonym or Index term are displayed in the Access window. Chapter 15 explains how to add a list of synonyms to take full advantage of the Look For feature.)

Guide Ideas

Following are more examples of content for Guides that you can create:

Server Access Quick 'n Easy

Most companies with multiple computers have one or more networks so that users can share information and resources. Usually such networks have servers that contain files and programs that staff members need. A Guide could help them navigate the information highways painlessly. For example, one task could lead users through the steps of creating an alias for a server volume or a shared folder so that they need only double-click (and supply a password, if necessary) in order to connect.

For everyone on a network

Ten Most Wanted

Every application program or work procedure has quirks and problems that challenge a number of users. Those puzzles—and their solutions—can make a very useful Guide. The customer support department for a program's developer could provide a list of the most commonly asked questions and answers to those questions, the content for a Ten Most Wanted database. The training or technical staff for a company or school could provide comparable information for the specialized procedures of the work group. And an enterprising designer could create Guides for a variety of these top-ten lists.

For most of us

Prices—Finding and Changing Them

Most businesses have computerized their order processing, invoicing, inventory, and similar collections of data. Guide databases can be especially helpful for new employees as they learn the system or for everyone when new software or procedures are introduced. For example, one task in a Guide could lead users through the steps of connecting to a server, opening a database, and retrieving prices for merchandise. Another task could supply instructions for updating prices in the database.

For all database users

EIGHT
ADDING INDEX TERMS TO THE GUIDE

After you've entered the panels for tasks and Topics, you can add the Index terms and assign content to them. First, however, you need to make a list of all terms and phrases that might be included in the Index. Then you need to decide how to use them.

Now also is a good time to write the definitions of the key terms for your Guide, if you haven't already done so.

Hands On: Building the Index

If you've listed the keywords as definitions in the Topics section of the Access window, you can begin by entering those terms in the Index. (For information about compiling the terms for the Index, see "Head Work: Listing Potential Index Terms" later in this chapter.)

Enter Index Terms

Follow these steps to add terms for the Index:

1. Open the application you created with Guide Starter.

   (If the application displays a Single-list Access window, you can't use an Index.)

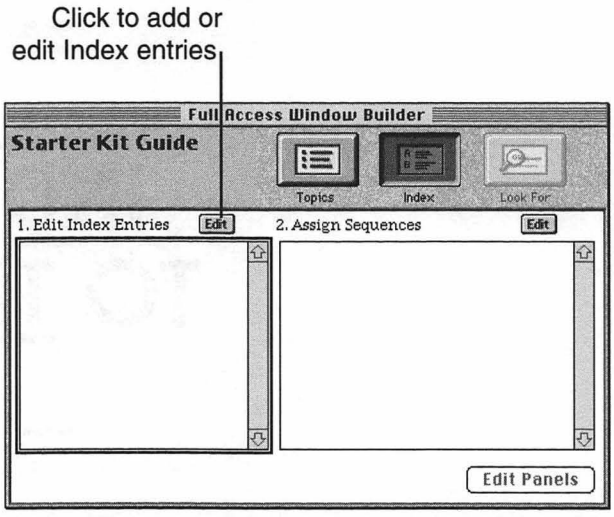


Figure 8-1
The Index section of the Access window

2. Click the Index button in the Access Window Builder.
3. Click the Edit button above the list on the left side of the window (Figure 8-1).

   The editing section appears at the bottom of the window when the Edit button is engaged.

4. Click to place an insertion point in the text box at the bottom of the window. Then type an Index term and click Add (Figure 8-2).
5. Continue entering Index terms until you've completed the list.

   After you add the second term, and for all terms thereafter, Guide Starter puts the entries in alphabetical order.

Assign Topics to Index Terms

After you've entered the Index terms, you can assign one or more Topics to each term.

Follow these steps to assign Topics to Index terms:

1. Select an item in the list of Index terms on the left (Figure 8-3).

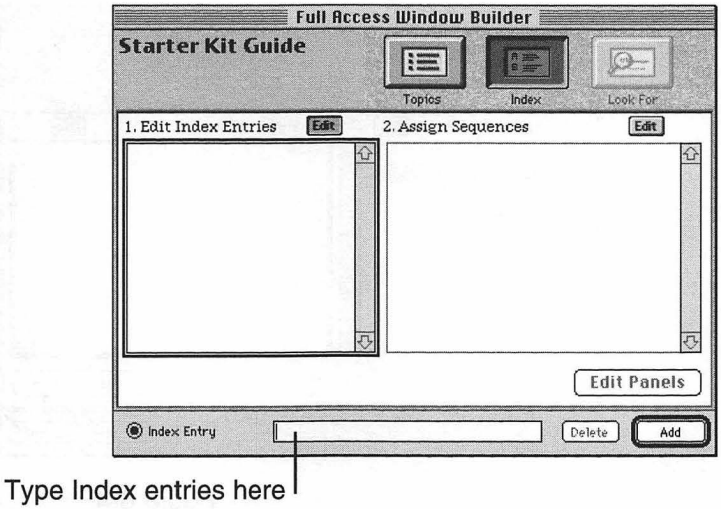


Figure 8-2
The editing section for Index entries

2. Click the Edit button on the right side of the window (see Figure 8-3).

   When you select the Edit button, the Index terms become dimmed and the items change in the editing section at the bottom of the window (Figure 8-4).

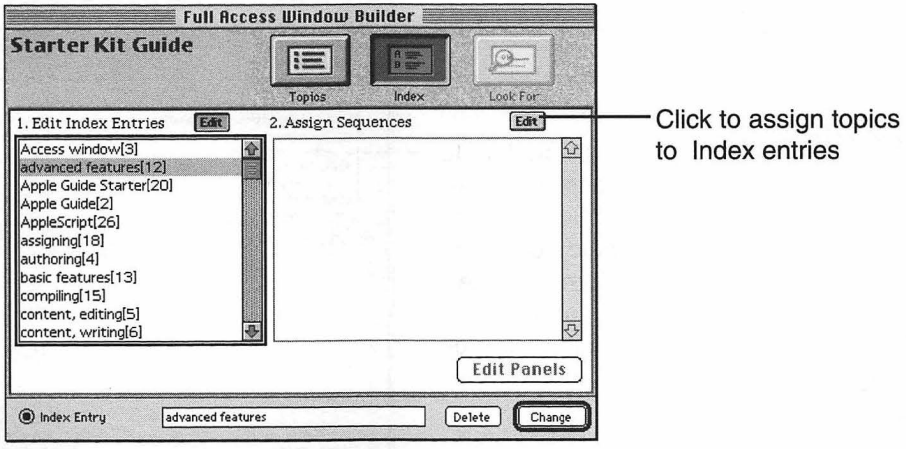


Figure 8-3
An item selected in the Index list

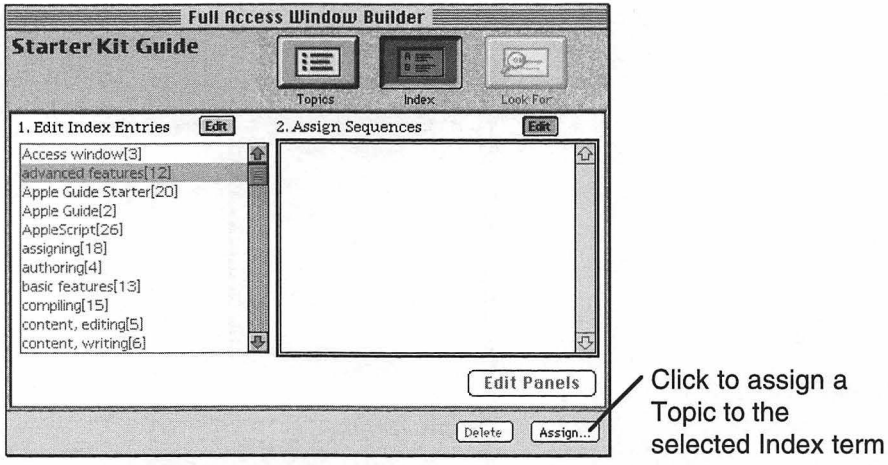


Figure 8-4
The editing section for assigning Topics to Index terms

3. Click the Assign button to assign a Topic to the Index term you selected.

   The Assign Sequences and Panels dialog box appears (Figure 8-5).

4. Select an item in the list.

   When a sequence is selected, the first part of text for each panel in that Topic appears (dimmed, but readable) in the Panels list (Figure 8-6).

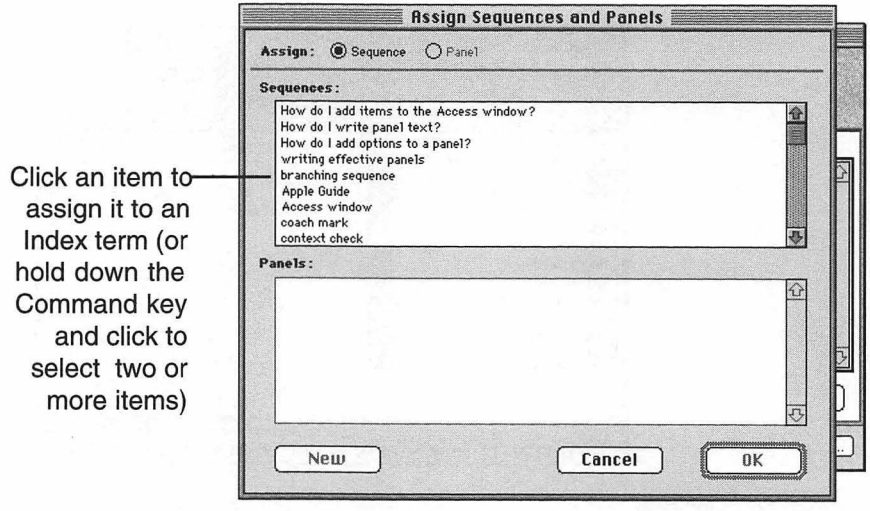


Figure 8-5
The list of Topics (or sequences) for assigning to Index terms

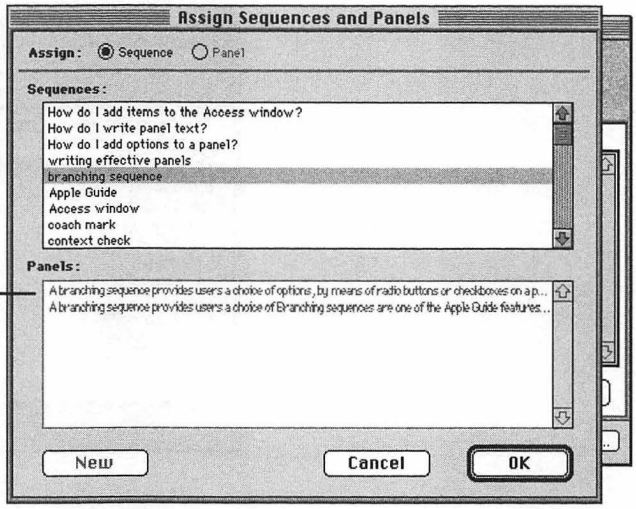


Figure 8-6
One sequence selected and its panels' text displayed

You can check the panel text to verify that the Topic selected is the one you want. Or you can hold down the Command key (⌘) to select two or more sequences. When you select the second one, the panel text in the lower part of the window disappears.

5. Click OK.

   The Topic and its Header appear in the list of assigned sequences, on the right (Figure 8-7).

In most cases, you'll want to assign more than one Topic to an Index term. To make multiple assignments, also called hits, hold down the Command key (⌘) and click each item you want to assign.

Changing the Order of Assigned Topics

✔ You can rearrange the items in the list of assigned Topics by dragging them (Figure 8-8). You can move Headers as well as Topics.

Removing an Index Assignment

✔ Follow these steps to remove an assigned Topic from an Index term:

1. In the Access Window Builder, select the Index term in the list on the left (see Figure 8-9).

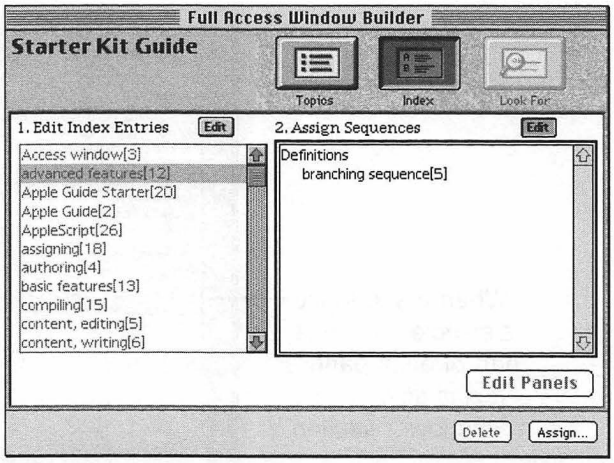


Figure 8-7
The assigned Topic and its Header are listed for the Index term

2. Select the assigned Topic from the list displayed on the right (Figure 8-9).
3. Click Delete.

Note: If you delete the only Topic for a Header, you must also delete the Header.

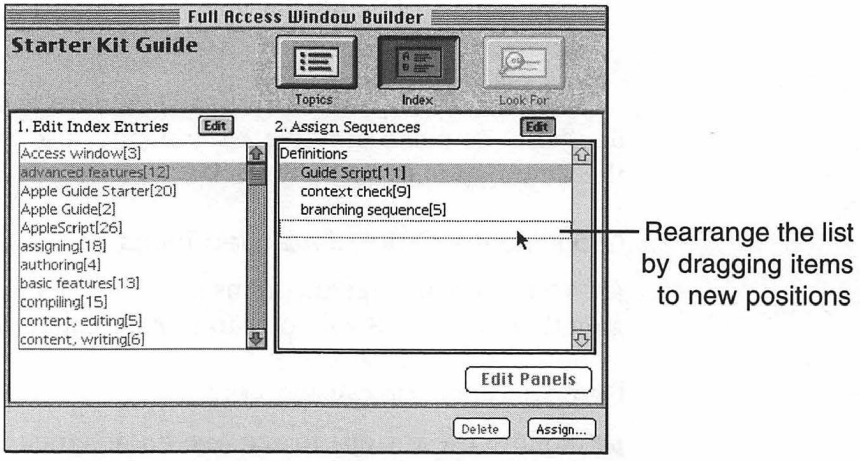


Figure 8-8
Topics and Headers in a list are easily rearranged by dragging

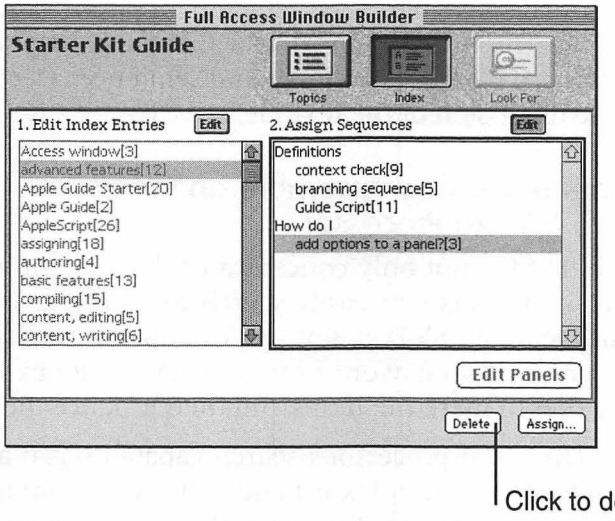


Figure 8-9
Deleting an assigned item from an Index term

Use the Text File for Index Terms

You can either go through the panels in Apple Guide Starter to study the content for key terms or have the program create a text file that you can consult. Using the text file is handy because you can print it out and highlight or otherwise mark the words you want to include. This gives you a reference source as well as a break from reading everything on the screen.

1. To create a text file of the content you've entered in Guide Starter, choose Build File from the File menu.
2. In the dialog box that appears, type a name (if you want to change Guide Starter's suggested name) and select a location for the file. Then click Save.

   Guide Starter builds the text file and puts it in the location you specified. This procedure takes only a few minutes. You can open the file with a word processor or with SimpleText (supplied with System 7.5).

In the text file, each element of Guide content is grouped together. The two sections that will be most helpful for locating Index terms are panels (section V) and sequences (section VI). For a complete discussion of the text file, see Chapter 11.

Mark Terms and Assemble an Index with a Word Processor

The text file not only concentrates the content you've already written; it also lets you easily search for content. Guide Starter creates this file in SimpleText, but you can open it with any word processing program. If your word processor has an indexing feature, you can use that to mark the terms and then produce the Index list.

The word processor's search capability can also be useful when you've made an Index list and want to be sure to assign all relevant Topics to a term. You can keep the text file open and search for each occurrence of that term.

Print the Text File or Open It for On-Screen Review

As noted, working with a printed version of the text file may help you avoid tiring your eyes. You can also always pick it up when you're away from the computer. If you don't use a printed copy, keep the electronic version open so that you can review it as you decide which topics to assign to Index terms.

Head Work: Listing Potential Index Terms

If you've used a template for keeping track of panel information or kept notes for tasks, you probably have identified many of the Index terms for your Guide. If you haven't kept a list of these terms, you can assemble one relatively easily by going through the content you've written.

The initial Index list should be as comprehensive as you can make it in order to ensure you include all relevant terms. As you review your notes and content, look for keywords, terms, and phrases associated with tasks and synonyms that users might think of for one or more Topics.

Keywords That Should Be Defined

Keywords are the primary terms for the product, such as the names of major activities users perform with the product. For example, if you're writing a Guide for a word processing program, keywords would include "text," "formatting," "editing," "printing," "fonts," "text styles," "saving," "documents," and so on.

In most cases, keywords are also the terms that should be defined in the Guide (unless the terms are commonly understood, such as text). You may have entered some or all of these words as Topics and written definitions at that time. If not, you can list these terms in the Index and later write definitions for them, which you do in the Topics section of the Access window. (This is a two-step process: You enter each term under its appropriate Topic Area and write the definition in the Panel Builder window. Then you move to the Index and assign each of the definitions to an entry.)

Task Terms and Phrases

Other terms or phrases are central to the Guide's content but don't require definitions. For a word processor, such words would include "inserting," "deleting," "tables," "word counts," and "margins." These tasks often correspond to the subsidiary tasks or operations users perform with the product.

You can often lift these terms or phrases directly from your Topic names. For example, a logical word processing task is "How do I find an item and replace it?" The Index entries for that task could be "find," "find and replace," "replace an item," "search," and "search and replace."

Synonyms and Related Terms

Ideally, an Index should provide broad access to a Guide's content. For this reason, it's a good idea to list as many synonyms as you can for the primary tasks and functions of the product your Guide covers.

Examples of synonyms for the search and replace task noted previously would include "locate," "look for," "substitute," and "overwrite." Even if you don't use these somewhat tangential terms in your Index, they are available for the Look For component (discussed later in this chapter and in Chapter 15).

If you are working with a computer support person or a group of users while creating your Guide, you can ask them to brainstorm Index terms and synonyms for you. Show them the list of tasks in the Access window (or a selection from that list) and ask them to tell you what words they'd look for in an Index to locate each of the tasks.

Decide Whether to Augment the Look For Feature

Because the Look For feature and the Index are related, you should be aware of this feature as you decide how extensive to make your Index. The Look For component allows users to type in a word or phrase and see what Topics are assigned to that text. The Apple Guide engine compares the user's phrase with three lists of governing terms (which you supply) and then searches the Index to see if anything there matches the search phrase. If there's a match, the Topics assigned to the matching term are displayed. If not, a message advises the user of this fact.

The Look For component is available in a compiled Guide even if you don't provide any listings for it. However, you can improve a user's chances of getting the desired search results by supplying the three lists of governing terms. These terms tell Apple Guide what words to ignore, what words not to shorten to their roots (or "stem"), and what words are synonyms for specific entries in the Index. Because Guide Starter doesn't include Look For in its automated features, you need to add the lists of governing terms in the text file (see Chapter 15).

It's advisable to add the governing lists in the text file if you can. This is because many users want to avoid the time and effort of reading the lists of Index terms or Topics, and Look For lets them quickly enter the term or phrase they want to find.

If you're not sure whether you'll be able to add the extra Look For content to the text file, enter your entire word list, including synonyms, in the Index so that all the terms appear somewhere in the content and have a Topic assigned to them. This should be sufficient for most Guides. However, you may want to test the Look For fea-

ture after you compile your Guide to verify that it finds the items in your Index.

If your Index is large, consider omitting the synonyms from it and adding them instead to the text file as part of the Look For content (see Chapter 15 for details). In this way, users won't have to scroll through or read so many disparate terms in the Index, and users who don't find the term they have in mind can go to Look For and perhaps locate it with a search.

Think Broadly

As you assign hits for each Index term, think expansively about what tasks and Topics users might expect to find listed for it. To make your Guide as comprehensive and useful as possible, be generous in assigning Topics to Index terms: when a Topic seems even tangentially related to the term, add it to the list of hits.

There are two possible exceptions to this highly accessible approach to assigning Index hits. One is a very large database (such as Macintosh Guide in System 7.5), in which the sheer size may require that you prune the lists of Index terms and hits. The other is a database whose design objective is extremely limited. In these cases, the liberal use of hits could be detrimental to the overall usability of the Guide.

Determine the Order of Items

Even if you've done a first pass at adjusting the order of hits for the Index terms, take another look at this part of the database. If several Topics are assigned to multiple terms, check the order and make it consistent. Apply the criteria for ordering items that you used to determine the order of Topics and Headers in the Topics section of the Access window.

Add "See Also" Items

Another way of providing broad access to the content of your Guide is to include cross-references to related terms among the hits for an Index term. This is something of a judgment call because the extent to which you use such cross-references should depend on the number and complexity of assigned items throughout the database. As always, your objective is to give users the easiest way to find and use the information they need.

Write Definitions for Keywords

If you've listed some keywords in the Index but have not written definitions for them, this is a good time to add those definitions to the database. You'll need to enter this content in the Topics section of the Access window.

Add Definition Assignments

After you've entered definitions for all the keywords, you can return to the Index section and complete the assignments for those terms. Follow the procedures in "Assign Topics to Index Terms" earlier in this chapter.

Check Assignments Again

After all assignments have been made, it's a good idea to review them one more time before you compile the database and release it.

Review the Sequence List

You can use your notes or the Guide's text file to remind you of Topics as you look at the assignments you've made for each Index term. The sequences section of the text file (number VI) shows all the tasks and Topics in the database.

Add More Hits

As you go over the assignments, you can easily add more hits to Index terms. As we've urged before, give users the maximum access to the instructions in your Guide by providing a rich Index.

Guide Ideas

Following are more examples of content for Guides that you can create:

Get Me Out of Here!

Many more Macintosh users than we might imagine are confident when doing tasks that are familiar to them but timid and frustrated when trying to master a new technique. Often they use a program's help or quickstart booklet to get into a template or a special format, and then they can't get back to their original format or document. For example, if a word processing user

sets the text format to a bulleted list and hits the return key for each new paragraph, all the paragraphs will have bullets preceding them. A basics-oriented Guide for that program could explain how to get out of such predicaments and also give general concepts that could rescue a user the next time. (In this case, the user could have simply copied a paragraph from the first part of the document—where no bulleted list appeared—and pasted it below the bulleted list, thereby resuming the original format.)
For relative newcomers to the computer or to a procedure

The Night Before Guide for Presentations

The entrepreneur who creates a no-frills, highly specific Guide to making slides in a presentation graphics program will win the gratitude of all the overworked people who have to prepare a big presentation in a bigger hurry. This kind of Guide can be very directive, telling users exactly what to do, which template to use, and so on. If it's feasible, the panels could also educate gently, explaining, for example, that the program has background and foreground layers and identifying each.
For all panicked presenters

Rapid Prototype

Apple Guide Starter is especially useful for creating prototypes or review versions of instructions. Instructional designers can try out a new approach, demonstrate a different style of coach mark, or get users' or engineering teams' feedback to a mini-Guide—before they prepare all of the content. Software designers can create prototype Guides to demonstrate how Apple Guide will be integrated with their programs. Both groups can make short show-off Guides to demonstrate their skills and ultimately can get jobs.
For instructional designers and software creators

NINE
BUILDING A TEXT FILE AND COMPILING A GUIDE

Once all the content is entered in your Guide Starter application, you're ready to create a text file and to compile the Guide. You can ensure that the compile process is efficient by making a few simple preparations and checking some key information as you proceed.

Hands On: Completing the Procedure

To compile your database, Guide Starter needs certain files and conditions. These are most likely already part of your system, but it's a good idea to verify that your Macintosh is ready for the compile procedure.

Run System 7.5 with at Least 8 MB of RAM

Your system software must contain certain of the standard components, including the Apple Guide extension and the AppleScript files, in order for Guide Maker Lite (the compiler program) to successfully build a Guide database. If you've removed items from your System Folder, check to ensure that Apple Guide, AppleScript, and the Scripting

Additions folder are in the Extensions folder inside the System Folder of your hard disk. If you use a Power Macintosh, the XTND Power Enabler file (a shared library) must be in the Extensions folder for Guide Maker Lite to run.

In addition, your Macintosh should have at least 8 MB of RAM for the compile process. Although the precise amount of memory used by system software varies by computer, System 7.5 occupies roughly 3 MB or more, Apple Guide Starter 1.25 MB, and Guide Maker Lite at least 1.6 MB.

Assemble the Necessary Files in a Folder

Guide Starter and its compiler program record the location of key files, such as a custom logo, and use those locations as the database is being compiled. To avoid problems with "lost" files, try to always keep your Guide Starter application in a folder along with all the files it uses.

The files that must be available for compiling include the following:

Mandatory

- your Guide Starter application
- Guide Maker Lite
- Standard Resources (from the Starter Kit disk)
- a logo art file (if you specified a custom logo in the Preferences)

Optional

- SimpleText (the text editor provided with System 7.5, which Guide Starter uses for the text file)

If you don't keep these files together or if you moved any of them after you began writing the database content, be sure to check the path information in the Preferences section before you compile (as discussed below). A file-location problem can also occur if you started work on the Guide on one disk or computer and then moved it to another disk or computer, even if you keep all the files together in a folder.

Check Guide Preferences

Before you build a text file or a Guide, go through the Preferences screens in your Guide Starter application to review the choices you made. With one exception, you can change the specifications for your database without negatively affecting the finished Guide. The exception is the Access window's style. If you change from Full to Single-list, or vice versa, some or all of your content will not be available. To review your choices, do the following:

1. Open the Guide Starter application for your Guide.
2. Choose Preferences from the Edit menu.
3. Verify that the options you want are selected and then click Exit.

Check All Path Information

As you review the Preferences, note the path information for each item you specified. These items include

- a PICT file for custom logo,
- a folder for the Guide, and
- the name (and location) of a program associated with your Guide.

You must write down or remember this information because you can't check in any folders or windows not already visible until after you exit the Preferences section of the program. If the path for any item is incorrect, then return to the Preferences and reselect that item.

Verify the Name for the Guide Menu

This is your opportunity to change the name of your Guide if you want to. As you're practicing with Guide Starter applications and building prototype Guides, you may want to change the name of the database each time you compile it. If you use a sequential naming scheme, you'll also have a record of the development of your Guide's content.

Another reason for changing the name of a database each time you compile it is to preserve all the variations. If you keep the same name (and location) for the Guide, the old version will be replaced by the new one whenever you compile.

Review the Logo Art Selection

You may have decided not to specify custom logo art because either you wanted the Guide Starter application to show the database title as you worked or you didn't have a logo to display. You can change that designation, either to add a custom logo, to change the art you want to use, or to remove a logo you previously specified. If you're adding a custom logo, be sure to check the box labeled "Use custom logo art" in addition to selecting a file with the Select PICT File button. (If the checkbox is not active, the custom logo will not be included, even if its name appears in the Preferences.)

Check the Greeting Text

This is also a good opportunity to proofread the Howdy text that users see when they open the Guide. You can assess whether you want to revise any of this text as you go over it.

Build and Check a Text File

After you've finished reviewing and making any changes you want in the Preferences, create a text file that contains all the content for your database. Your Guide Starter application adds the correct Guide Script syntax and tags to your content and then saves the file as a SimpleText document.

Follow these steps to create the text file:

1. Choose Build File from the File menu.

   The directory dialog box appears.

2. Type a name for the file (if you don't want the one preset by Guide Starter), select a location, and click Save.

   The text file appears at the location you specified.

It's a good idea to look over the text file before you compile your Guide. See "Head Work: Becoming Familiar with the Text File" later in this chapter for more information about the text file.

Compile the Guide

When the content is complete and correct, compile your Guide.

Follow these steps to compile with Guide Starter:

1. Choose Build Database from the File menu.

   The directory dialog box appears.

2. Type a name for the text file (if you want to change the suggested name), select a location, and click Save.

   Important: The text file must be in the same folder as the other files required for compiling—Standard Resources, logo art (if used), Guide Starter, and Guide Maker Lite.

   When you click Save, the dialog box closes, and Guide Maker Lite opens in the background. A message appears on the screen that shows the progress of compiling with a bar graph (Figure 9-1).

After a few moments (depending on the size of your file and the Macintosh model you're using), the procedure is complete and an alert message indicates whether compilation was successful.

Check the folder or disk you selected for saving the Guide. You should find the Guide along with the text file and a log file (a SimpleText file that has the same name as the Guide with ".err" at the end).

Occasionally you may see an error message instead of the compile status dialog box. Take note of what the message says. It's possible that not enough memory was available for both Guide Starter and the compiler program to open at the same time. Check to see if other programs are open and if so, quit them and try to compile again.

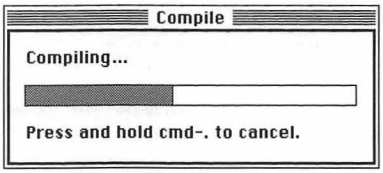


Figure 9-1
The status bar displayed by Guide Maker Lite

Consult the Log

Open the log file for your Guide and read it. In most cases, it will report that the compile has been successful.

If the compile was not successful, the alert message at the end of the procedure asks if you want to open the log. Click OK when you see that message, or open the log later. This record of compiling states what part of the text file had a problem.

Figure 9-2 shows an example of a log from a compile that was not successful.

In rare instances, you may encounter another type of problem. The log file should give you adequate information to resolve it.

Head Work: Becoming Familiar with the Text File

You probably noticed the Build Database command just below Build File in the File menu. Though it may be tempting to go ahead and use that command (which builds the text file and compiles it into a database), take a few minutes first to look at the text file. Doing this gives you one more chance to catch any errors or identify items you want to change.

Equally important, becoming familiar with the text file is a gentle introduction to the Guide Script language and the basic structure of a scripted program.

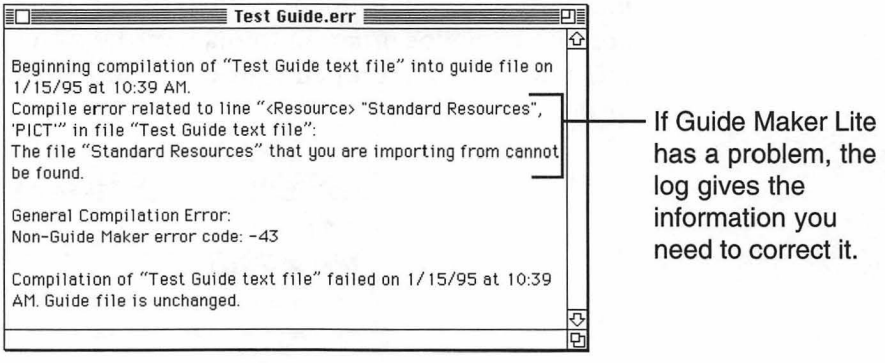


Figure 9-2
Example of a log produced by Guide Starter's compiler program

Print a Reference Copy with Page Numbers

You can reduce the amount of screen reading you have to do and get a useful reference document by printing the text file. You can open the file either in SimpleText by double-clicking the File icon or in a word processing program by launching that program and using its Open command to open the file. (SimpleText can't open files larger than approximately 32 K, so you must use a word processing program to work with a text file larger than that.)

If you use a word processor to print the text file, be sure to add page numbers before printing. SimpleText provides them automatically when you print the file.

Use the Numbered Sections for Easy Access

As you'll note when you study the file, the text is divided into twelve sections, each with a number and heading, so that you can easily find the parts of your content you want to check. The first eight sections cover the Apple Guide features that Guide Starter automates. The last four sections are provided for the advanced features (covered in Chapters 13, 15, 16, and 17), which you can add only in the text file.

Figure 9-3 shows a portion of the text file for the Starter Kit Guide.

Another reason Guide Starter divides the text file into numbered sections is to let you easily locate each type of content with the search feature of a word processing program.

Begin Learning Guide Script

Studying the text file is the best way to begin learning the Guide Script tag language that Guide Maker Lite needs to create a database with your instructional content. As you work with text files for several stages of writing and revising a Guide—or many Guides—you'll start to recognize the correct format for various elements in the database, and you'll be able to copy or adapt parts of a text file for use in other parts of the Guide or for new database files.

Check Vital Statistics

Look over the first part of the text file carefully. This is where the Guide's title, logo name (if used), and Howdy text appear. It can't hurt to check them one more time.

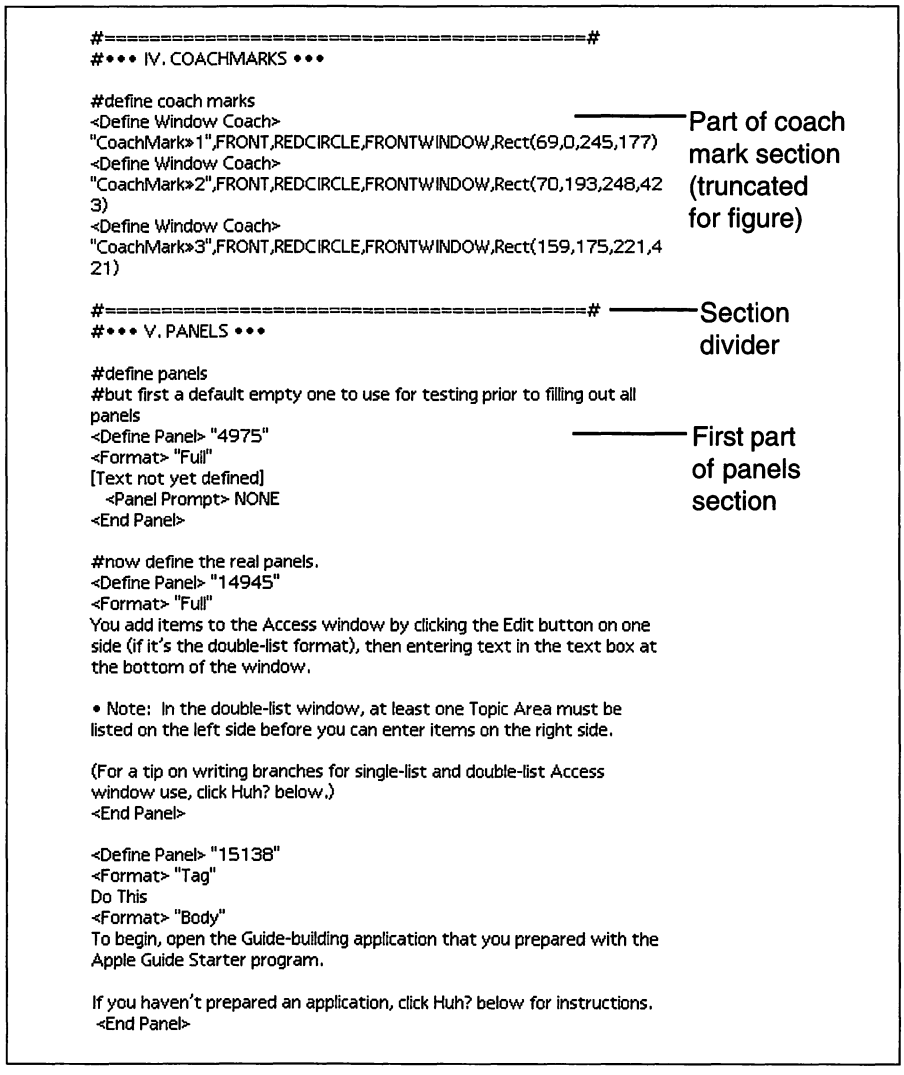


Figure 9-3
Part of the text file for a Guide

Check Spelling with a Word Processor

Even if you printed the file with SimpleText, consider opening it in a word processor so that you can use the program's spell-checking component to review the file. If the spell-check process locates any errors, note them on the printed copy of the text.

Make Corrections in Guide Starter

If you need to correct any spelling errors, go back to your Guide Starter application and make the changes there. Changes made in the text file won't be in Guide Starter unless you enter them there as well. And if you compile the Guide Starter content without making the corrections in its content, the errors will be included in your Guide.

(You can make changes in the text file and then compile it by using the Build Database from File command. But you'll have to either go back to your Guide Starter application later and add the corrections there or continue working only in the text file.)

Troubleshooting and Correcting Problems

The most common cause of a compile failure is a missing file. For example, if you save the Guide in one place (you designate the location for the Guide in the Preferences of your Guide Starter application), save the text file in another place, and have Guide Starter and Guide Maker Lite in a third place, the complex location scheme can cause a problem.

Important: Guide Starter and its compiler require the text file to be in the same folder as the crucial Standard Resources file. To avoid fooling Guide Starter and its compiler, save the text file in the same location as Standard Resources and the two Guide Starter programs.

Make Backup Copies

After you've compiled your Guide, be sure to make copies of it and of the text file and put them away so that you don't use them for regular work. You could find them very useful in the future if you need to trace the changes in a Guide during its development.

Use an Informative Naming Scheme

When you make a copy of each version, give it a name that tells you the key details about it. For example, you could include in the name a version or phase number, the current date, and even the time. You can add more information to the copy if you want via the Comments section of the Info window for the file. (To add comments, select the file's icon and choose Get Info from the File menu.)

Using informative names for backup copies is particularly important if you don't change the name of your Guide each time you revise and recompile the database. Having a clean copy of each version you built may save you lots of work should anything go wrong with the production version.

Establish an Archive

It's easy to lose files that show the history of your Guide's development as you continue to create new Guides and as more files compete for space on your hard disk. You can ensure you have copies of all your work—and provide a record for anyone who may need to take over for you or collaborate with you—by creating an archive of all your important files and updating it regularly.

Guide Ideas

The following are more examples of Guides that you can create:

Palette Practice

Graphics programs usually include palettes of tools, patterns, colors, textures, and other image enhancements. Many users have only a casual acquaintance with their graphics programs, so most would benefit from a Guide that explains the purpose of each item on the program's palettes and also provides exercises for practice in using them. If it's feasible (depending on disk space and RAM available), you might even include sample images that users could try to match and also copy and modify. (Chapter 13 explains how to add graphics to the text file for a Guide.)

For everyone who uses a graphics program

Extensions Explained

Strange things sometimes happen to the computer. One source of mysterious problems can be conflicting extensions—essential little chunks of code that most users know little about. A savvy Guide could explain what extensions are, offer tips and remedies to help users avoid problems caused by unfriendly extensions, and perhaps even launch a utility program that can keep extensions from causing havoc with the system.

For everyone

Spreadsheet Modeling Guide

Spreadsheets are versatile but complex tools that require study and experimentation for people who don't specialize in their use. An informative Guide database could fill the knowledge gap for these users by providing step-by-step instructions for building templates to track retirement funds and other investments, record income and expenses for taxes, and similar calculation tasks. The Guide could use coach marks to illustrate how to create formulas.

For anyone who's not a spreadsheet wizard

TEN

TESTING AND REVISING YOUR GUIDE

After you've compiled your Guide, try it out and, when you're ready, distribute it to reviewers or volunteer testers who can help you catch any errors or holes. The mechanics of testing—and later revising—your database will vary according to its purpose and audience. But some common-sense strategies can assure that you get useful feedback and manage the revision process successfully.

Hands On: Checking and Improving Content

The testing and revision stages are as important to your Guide's success as the research and writing stages. The steps you should take include the following:

1. Go through all tasks in the Guide while using the product.
2. Observe users as they work with the Guide and the product.
3. Revise the Guide.

Test the Guide Yourself

You're the expert for this Guide because you designed it and you've worked with the product to develop the tasks and instructions. When you

compile a database for the first time, give it a thorough review before you release it to others.

Go Through All Content with the Product

For a new Guide or one that you've revised substantially, use your instructions to work through all tasks with the product. When you've made only limited revisions, you can concentrate on just those Topics (assuming you've previously reviewed all content).

Try All Variations on All Configurations

If you've included branching tasks, context checks, or other conditional content, be sure to set up your system to check each situation. Similarly, if the instructions in your database cover several Macintosh models or multiple programs, try to use the Guide on each different configuration.

Use the Text File

For the mechanics of checking your Guide and managing the changing content, print the text file that the Guide Starter Compiler used to build the database. In so doing, you can be sure that this collection of data matches what's in your Guide. You also will have a printed copy with you to work on even when you don't have access to a computer.

Make a Correction Copy (So That the Original Remains Intact)

As you proceed through a review-and-revision cycle, you could be juggling hundreds of details about the product and the instructions for it. So it's important to keep reference copies of each stage of your Guide's development. In this way, if you encounter a problem, you can check your versions and find out when it was introduced.

To ensure that the original text file remains untouched, make a copy of that file and use the copy as your reference source.

Add Version Number, Date, and Page Numbers

When you copy the text file, add a version number (use any scheme that works for you), the date you began your review, and any notes that you think might be useful later. For convenience in working with the printed document, add page numbers (if they aren't provided automatically by your word processing program).

Separate Text File Sections

You may also want to insert a page break between sections of the text file so that each subdivision of the Guide's content begins on a new page. Having a separate batch of pages for each section of the file is especially desirable if other people work on the Guide with you. For example, you could take the panels section and make your corrections in it while someone else works with the sequences.

Use the Change History for Reference

If you've revised the database previously, you should have a change history that notes all parts of the content that were revised. The change history can be very useful in helping you find the source of problems that crop up. Of course, it also reminds you which topics were revised since you last reviewed the content.

(See "Keep a Change History" later in this chapter for more information about a change history.)

Make Interim Corrections and Recompile If Appropriate

If your review uncovers instructional errors, misplaced coach marks, or other errors, you probably will want to correct those mistakes and recompile the database. Your opportunity to review and make corrections will depend on the schedule for the project and on whether there are eager users waiting to try out the Guide.

If you do make corrections, be sure to keep archive copies of the original version of the text file and Guide and of your corrected version.

Have Users Test the Guide

In addition to reviewing your Guide yourself, it's essential that you arrange for other people to test it. Although you're the expert, you are also involved in the details of the product and the database. So you can't look at the Guide with fresh eyes. (Neither can the product's designers or others on a team associated with the project. Like you, they are too familiar with it. But their feedback can still be helpful for revising.)

Ideally, you need the kind of spontaneous reactions to the database that someone with fresh eyes will offer.

Find Users from the Guide's Intended Audience

Getting reviewers (or testers) usually isn't difficult, at least for the first round or two of testing. If you're developing a Guide for a specific group, some group members will likely be willing to use the database and give you their feedback.

If you don't have access to the actual users of your Guide, try to enlist volunteers whose computer experience and work objectives are close to those of the intended audience.

Observe the Users as They Work with Your Guide

If you've had no experience with usability testing, this suggestion may sound odd, but it works: Set up a Guide-testing "lab" so that you can observe the users as they work with the product and your instructions. You can keep this testing environment informal—no videotaping or audio taping, for example—and still get extremely useful information about your Guide.

Standardize the Setting and Tasks

Basically, such a testing situation involves getting volunteers to, say, come to your office or meet you at a computer lab and let you look over their shoulders as they use the Guide. If you do make such an arrangement, try to get at least three people to go there separately and then ask them each to do the same set of tasks, using the Guide for their instructions. This way, you'll create conditions that are as similar as possible for each tester, and you'll see them reacting to the same parts of the database.

Although it may seem preferable to give the users different content to cover, by having them do the same tasks you increase the probability that their questions and problems will show you the weak places in your instructions.

Note Questions, Problems, Comments

Whether or not you can get volunteer testers to let you watch them work with the Guide, be sure to talk with everyone who does test your database. Do this as soon after the time they used it as possible. If you do observe your testers, ask them to basically think out loud, especially when they have a question or aren't sure what to do.

You're there to find out whether your instructions are working, so try not to answer the testers' questions or direct them to the right part of the Guide (unless they're really stuck). Instead, note their questions or problems and encourage them to give you their reactions.

Trust Users' Feedback (and Your Eyes and Ears)

Often it's tempting to dismiss someone else's comments—positive or negative—if they don't mesh with your estimation of a situation or product. This is why you want several people to go over the same tasks in your Guide: If they all have some similar questions or comments, you'll have a hard time ignoring their feedback.

Ask for Suggestions

If you do get a reaction to your Guide that is not positive, try to get the users' ideas for improving the parts of the instructions that gave them trouble. Users frequently can tell you how they would solve a problem creatively or how another product they've used explains a similar concept.

Look for Trends

The workable suggestions you get from users can be a nice bonus from this testing process because you want to find any trends in the problems, confusions, or successes the users had with the database.

For example, if you have five volunteers who do the same tasks, and three of them stumble over a description or pick the wrong Topic before finding the one you intended them to use, you very likely have a trend. (If they all sail through the tasks and instructions without problems, that's a trend, too.)

If four of the five testers generally have success, but the fifth has a terrible time, that is not a trend. You can probably get some useful suggestions from the user who struggled, but you're looking for a coincidence of responses to tell you whether your Guide is working. The four successes are your trend in this case.

Decide What to Change

Whatever type of testing or reviewing you are able to arrange, gather as many responses as you can, while leaving yourself time to revise before the next version is due.

Analyze All Feedback

Use your observation of users and any other feedback to help you answer a series of questions about the Guide. These questions might include the following:

- Do users accomplish tasks with the Guide's instructions?
- Are the Topic Areas, Topics, and terminology clear to users?
- Are there tasks or design elements with which two or three users had a problem?
- Did users offer opinions or suggestions that could improve the Guide's instructions?

Add other questions to this list according to the objectives of your database.

Evaluate the Design in Light of Problems

Consider your answers to the questions above, along with any problems or trends you observed, as objectively as you can. If users reported problems or confusion, determine whether the difficulty resulted from the design of the instructions. If so, you probably should look carefully at the Guide's design and perhaps ask others who have experience building databases to help you modify the Guide in order to avoid the problems users had.

If you do change the Guide's design substantially, make a prototype of the new version and have two or three users work with it so that you can verify that your revisions have eliminated the problems.

Identify All Factual Errors

Most often, what you'll get from reviewers and testers are lists of the little mistakes that can creep into any collection of facts, for example, the panel that says "Import Picture" when describing a menu item that's actually "Get Picture." Mark all such errors on your copy of the text file so that you have one master document for corrections.

Plan Your Revision

Finally, before you begin making changes in a copy of the Guide Starter application, devise a plan for your revision. If you'll be making changes in the structure of tasks, for example, you'll probably want to do that before making any other changes in the content. Otherwise you risk undoing some of your corrections when you restructure the content.

Similarly, if you need to rewrite some Topics to make them consistent with the rest, it's a good idea to do that as a separate procedure. Then you can go back through the database to fine-tune the phrasing and correct the scattering of small mistakes in panel text.

Revise Your Guide

Once you've identified the necessary revisions and decided how you'll proceed, you can dive into the details once again.

Work in Guide Starter

Unless you've made major additions or changes in the text file, use your Guide Starter application to revise your Guide. Doing this lets you continue using the program's automated scripting and Guide-like interface.

If you have included some of the advanced Apple Guide features in your text file, you probably should make your changes there. You can still use Guide Starter to compile the edited text file. Chapter 11 provides a full discussion of working with the text file.

Make a Copy

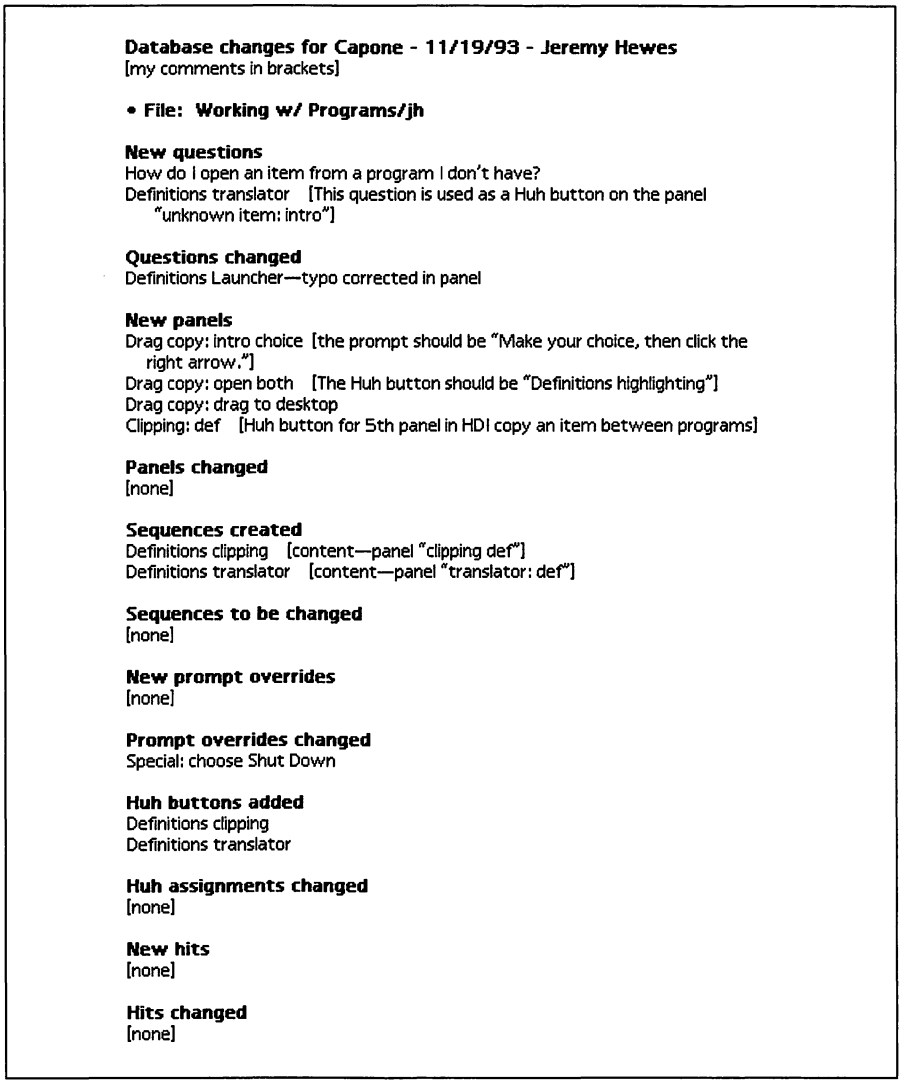
As with the text file you use to note corrections, make a copy of your Guide Starter application and use it for revising. Then you'll always have the original intact in case your working files are damaged or you can't determine what has been changed already.

Because Guide Starter saves automatically, you can't bail out on a big mistake by closing the file without saving your changes. So keeping a clean original is even more important.

Protect a Master

To ensure that you'll always have a workable copy of the most recent version, make a master and protect it by storing it on a locked disk or in a safe place away from your regular work location.

You may want to set up a folder or network server volume to store the original Guide Starter application and text files. If other people have network access to that location, you can use file sharing's access privileges to determine who can see or change your Guide files.



Database changes for Capone - 11/19/93 - Jeremy Hewes
[my comments in brackets]

• **File: Working w/ Programs/jh**

New questions
How do I open an item from a program I don't have?
Definitions translator [This question is used as a Huh button on the panel "unknown item: intro"]

Questions changed
Definitions Launcher—typo corrected in panel

New panels
Drag copy: intro choice [the prompt should be "Make your choice, then click the right arrow."]
Drag copy: open both [The Huh button should be "Definitions highlighting"]
Drag copy: drag to desktop
Clipping: def [Huh button for 5th panel in HDI copy an item between programs]

Panels changed
[none]

Sequences created
Definitions clipping [content—panel "clipping def"]
Definitions translator [content—panel "translator: def"]

Sequences to be changed
[none]

New prompt overrides
[none]

Prompt overrides changed
Special: choose Shut Down

Huh buttons added
Definitions clipping
Definitions translator

Huh assignments changed
[none]

New hits
[none]

Hits changed
[none]

Figure 10-1
A change history for a partial revision of Macintosh Guide

Keep a Change History

The best way to track the successive changes as you develop a database is to keep a change history that notes the specifics of all changes in content and scripting. Anyone who makes changes in a Guide's content or scripting should create such a document, and all change histories for a database should be kept together with the master copy of the Guide Starter application and text file.

Figure 10-1 shows an example of a change history for a very limited revision of Macintosh Guide.

If you are the sole author of a Guide, the change history is mainly for your use, although the clients for your database (if you were hired to create it, for example) might also want copies so that they can make future revisions with a knowledge of what's already been done.

Write Release Notes

If you distribute the Guide to reviewers during development, include a document containing release notes that reviewers should know about. Usually release notes report the changes in the product and the Guide, but not in as much detail as in a change history.

Repeat If Possible: Revise, Test, Revise

After you've made your revisions and recompiled your Guide, get volunteers—new ones or the same people—to review it once more. If the schedule (and budget) for your project permit, you should have at least two cycles of testing and revision before releasing the database. More testing and feedback is even better.

Guide Ideas

Getting Good Video

Capturing and editing video clips can be a bewildering experience, even when the hardware and software to do it are included with the Macintosh. All the recent arrivals to the world of multimedia could benefit from a Guide that presents the basics of video capture and editing, with QuickTime clips to illustrate the good, the bad, and the ugly.

For video newcomers

The Telecommuter's Survival Kit

The ranks of telecommuters are increasing rapidly, accompanied by an explosion of modems and other high-tech links between home and office. The folks who compute to work often have their highway headaches replaced by technology woes as

they try to get their e-mail remotely or set up the fax modem to answer on the thirteenth ring. These reluctant pioneers could really use a Guide to coach them through remote access, phone and message management, net navigation, and other mysteries of telecommuting.
For stay-at-homes everywhere

Your Guide of Guides

Guide authors and Guide consumers could benefit from a database that introduces the many styles of instruction and information delivery that Apple Guide has spawned. Consultants, for example, could make samplers of their various Guides to introduce themselves to potential clients. Training departments could pull together a Guide of effective and not-so-effective examples of customized instructions—and thereby eliminate some of the training they'd otherwise have to do.
For all Guide users

PART III

ADDING BELLS AND WHISTLES

ELEVEN
WORKING WITH THE TEXT FILE

As you know, Guide Starter builds a text file that includes the content you've entered in your application, along with the Guide Script commands the compiler uses to turn the file into a database. You can use this text file to add advanced features of the Apple Guide technology to your Guides. You can also learn scripting fundamentals as you work with this file.

The following considerations should be useful in helping you decide whether to add extra content in the text file:

- Context checks and AppleScript scripts are important for any tutorial or other Guide for novice Macintosh users.
- Coach marks that use AppleScript scripts or alternative commands are similarly important in a Guide for novice users.
- Pictures can be especially useful in a Guide if coach marks don't work with some parts of the product.
- A QuickTime movie or animation can demonstrate an action that coach marks can't support, such as inserting a floppy disk or a CD in a disk drive. These moving images are also good for demonstrating a product's features or showing a procedure on the screen.

- Branching tasks that use decision panels are efficient and reduce the total number of Topics in a database.
- Look For lists—synonyms, exceptions to the built-in stemming of Index terms, and words to ignore when searching—are particularly useful if the Guide is intended for experienced users, who frequently use the Look For access method because they know what they're looking for.

Evaluate Trade-offs

There are trade-offs in adding or changing content in a Guide Starter text file. Obviously the database and its users will benefit from the enhancements, and you will gain some valuable experience working with the Guide Script text.

At the same time, once you modify the text file and compile your Guide from that version, you must leave Guide Starter behind. If you prefer, you can continue to write basic content in Guide Starter, build a text file, and then copy the modifications from your previous text file to the newer one.

You can best evaluate the benefits of adding advanced features if you work with the text file, at least to a limited extent, and see one or more of these enhancements in your database.

Study the Script

As we've noted previously, the text file is simply the instructions you've entered in Guide Starter and the Guide Script commands, or tags, the compiler uses to convert the content into a database.

If you study this file, you'll see that it has an organization beyond the division into a dozen sections (which Guide Starter does for your ease of use). The content in the text file also conforms to the following rules:

- Every element in a Guide—panels, prompts, windows, text, font, and more—must be defined and given a name. The name can then be used whenever that element appears (and it can appear many times).
- Every panel must be part of a sequence.

- Every sequence must be assigned to at least one Topic.
- If there is even one missing character in script syntax, the text file won't compile; instead, an error will occur.

Keep these requirements in mind as you work with the text file so that you can avoid errors when compiling.

Learn Scripting

Another reason to try out some of the enhancements explained in this chapter and in Chapters 12 through 17 is to become familiar with the format and language of Guide Script. The more experience you have in devising ways to use AppleScript, context checks, branching tasks, and other features, the more versatile and confident you'll become at building Guides.

After you've worked through the examples in this chapter and the rest of Part III, you'll be ready for *Apple Guide Complete*, the Guide-authoring handbook for programmers and software developers.

Improve Your Content

Modifying and adding features to the text file can also help you improve the instructions in your Guide. As you work in the text file, you'll see your content from a new perspective. You may see ways to make the task structure more efficient or to make instructions more consistent among tasks.

In fact, printing the text file and reviewing it is an excellent way to assess the completeness and accuracy of your Guide, even if you don't add advanced features.

Examine the Text File

The text file Guide Starter creates contains twelve sections (eleven for a database in the Single-list format). The information you supplied makes up about half of the file's content; the rest is scripting or comments that describe the scripting.

The figures that follow show the portions of the text file for a Guide that contains two Topics. This text file is for a database named Template-making Guide and is included on the Starter Kit disk.

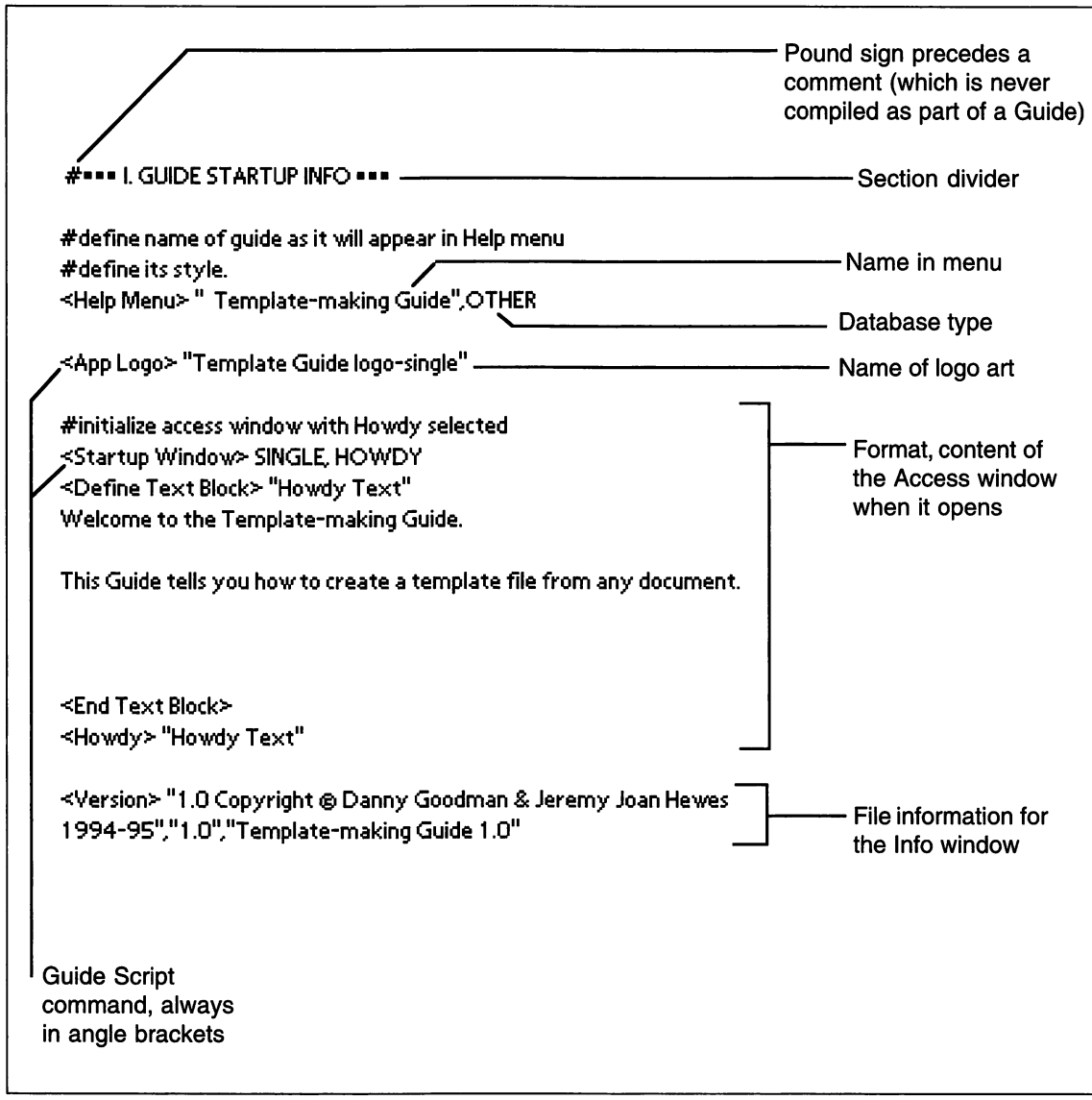


Figure 11-1
The first section of the text file, showing basic information about the Guide

At the end of this chapter, and as the last section in some of the other chapters in Part III, we provide examples of advanced features that use the Template-making Guide's text file. (To make the text and labels fit on the book page, we've changed the font used in this text file. However, the content is the same as in the copy on the disk.)

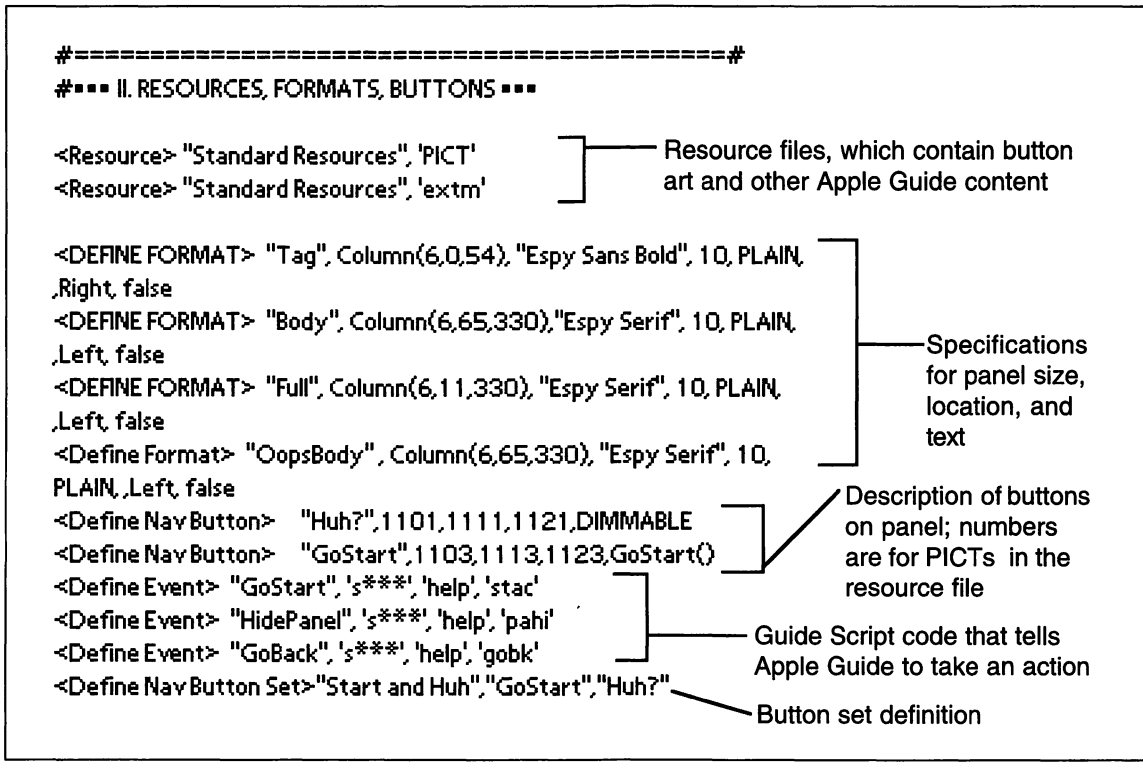


Figure 11-2
Descriptions of panel's text area and buttons on the Presentation window

Figure 11-1 shows the first section of the text file. This section defines the Access window and type of Guide, names the Guide, and designates the resource files the compiler will use for visual elements such as buttons and a logo.

Figure 11-2 defines more basic elements of the database, including panel size and font, buttons on the Presentation window, and actions that the buttons initiate.

Figure 11-3 shows two sections of the text file—the prompts and coach marks. The prompts in this file consist of Guide Starter's standard set (displayed if you activate the Prompt checkbox in the Panel Builder window) and the custom prompt "-- End --." The two coach marks in the file both indicate items in the File menu.

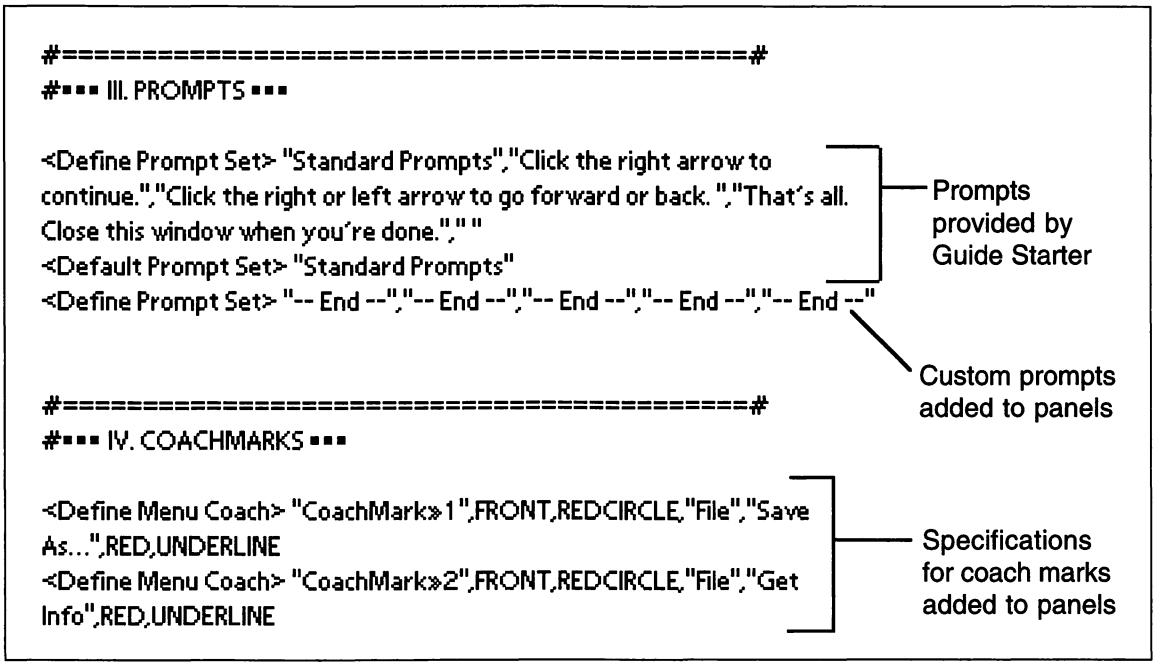


Figure 11-3
Prompts and coach marks for the Template-making Guide

Figure 11-4 shows the first part of the panels for this text file. Notice that the first panel is a default panel provided by Guide Starter.

Just as Guide Starter provides a default panel, the program also creates a default sequence to supply placeholder content for any Topic that has no panels. This database contains two Topics (besides the default). One has one panel (a definition), and the other seven (a task), as Figure 11-5 shows.

Figure 11-6 shows the Access window content in the text file. As with panels and sequences, Guide Starter provides a default Topic Area.

The four sections shown in Figure 11-7 are empty, except for a comment, because they are designed for advanced features not included in Guide Starter.

Section X for AppleScript scripts won't contain the Guide Script commands used to call those scripts. Rather, the commands are used in the appropriate section of the file according to their content, for example, coach marks using AppleScript scripts in Section IV.

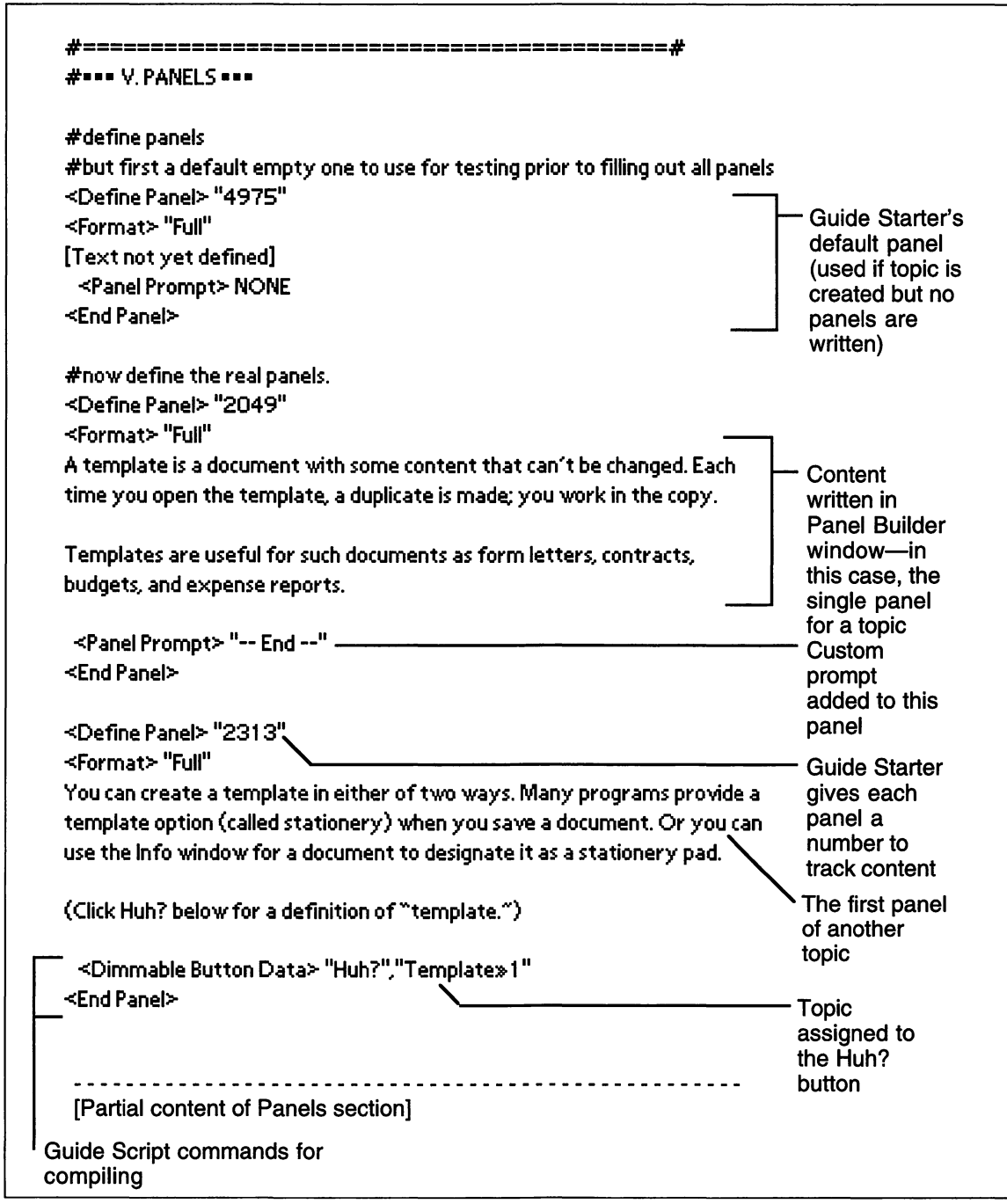


Figure 11-4
The first few panels in the text file

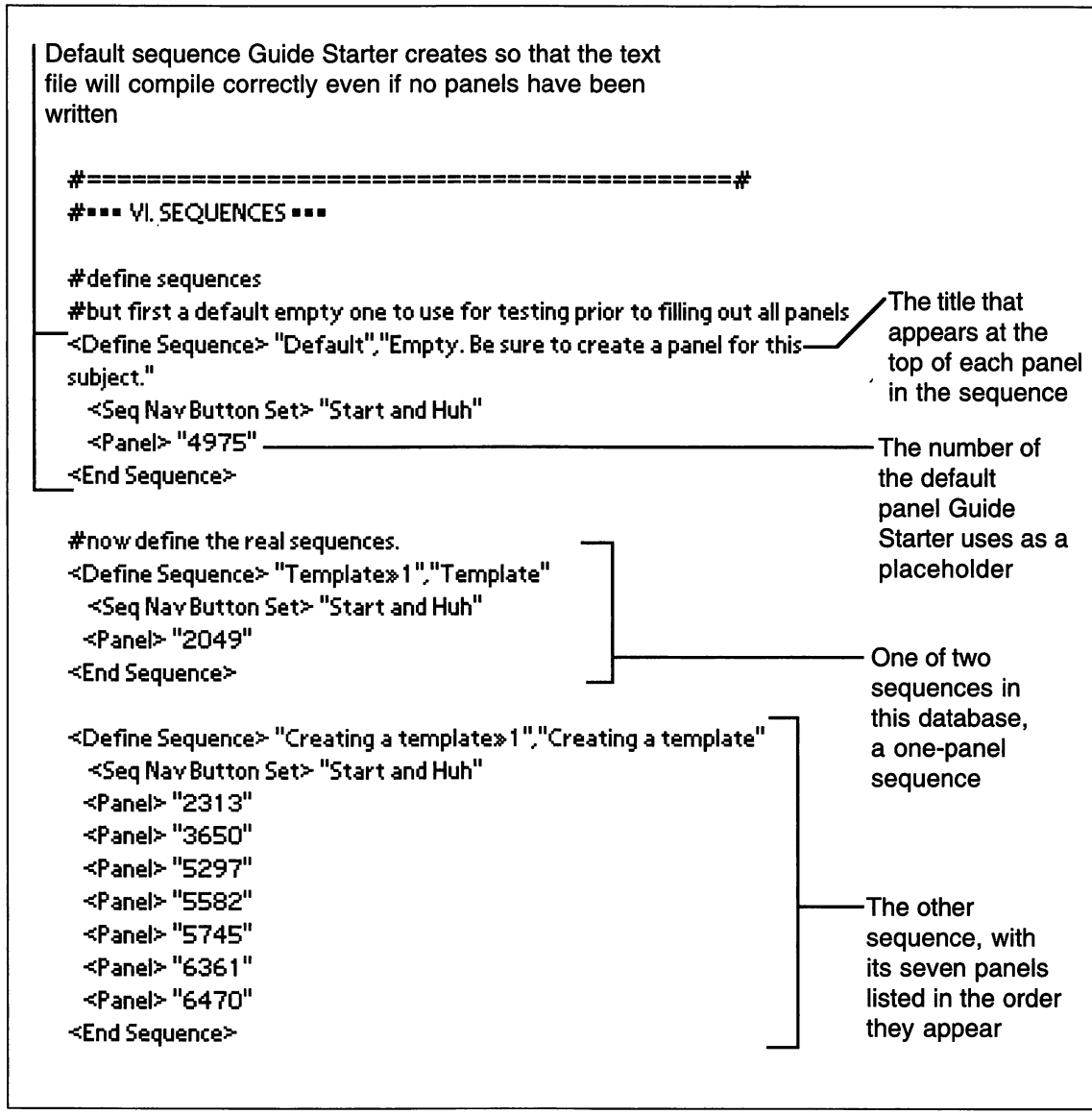


Figure 11-5
The sequences section of the Template-making Guide text file

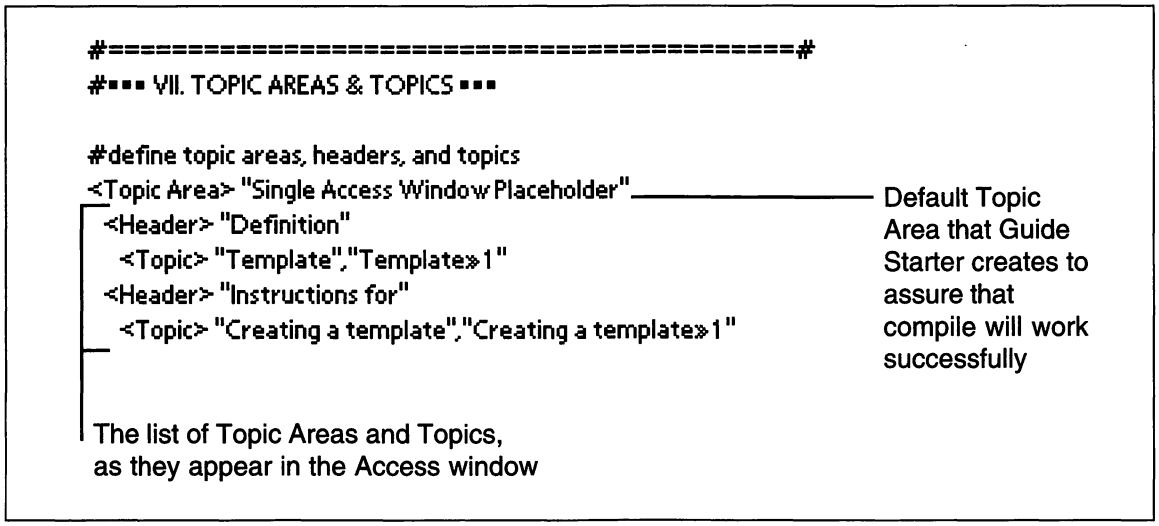


Figure 11-6
Headers and Topics in the text file

When you begin to add features in the text file, you'll probably want to print the most recent version for reference.

Experiment with New Features

As noted, you'll have several chances to modify and add to the text file in this chapter and the next several chapters. The advanced features covered include the following:

- changing panel numbers to names
- adding a new button on the Presentation window
- changing the style of a coach mark (Chapter 12)
- adding a different type of coach mark (Chapter 12)
- changing the size of the Presentation window (Chapter 13)
- inserting a graphic on a panel (Chapter 13)
- adding a QuickTime move to a panel (Chapter 13)
- modifying a task to include two branches (Chapter 14)
- creating word lists to supplement the Look For feature (Chapter 15)
- using an AppleScript script to locate and open a document (Chapter 16)
- using a context check in a task (Chapter 17).

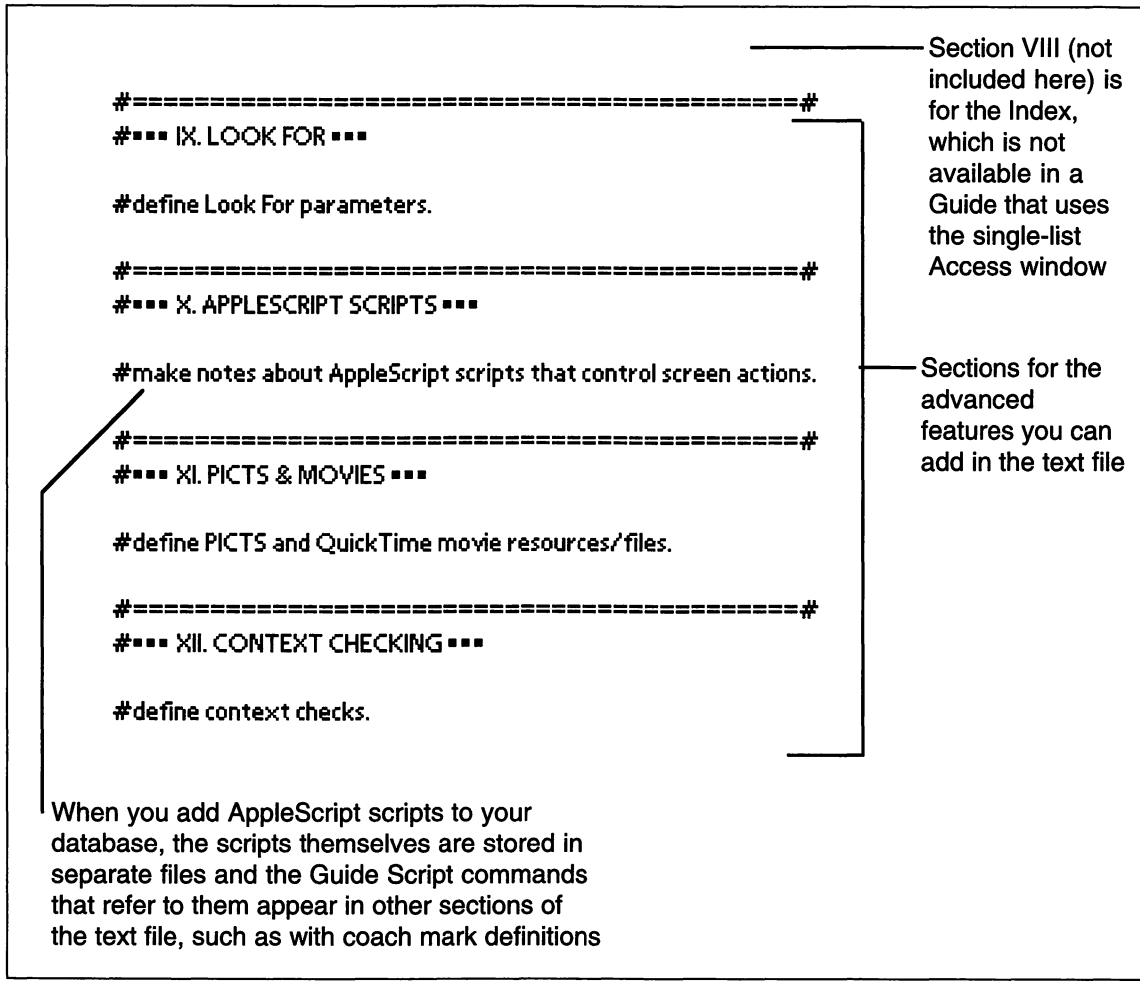


Figure 11-7
The four sections for scripts that provide advanced features in a Guide

You can use the disk copy of the Template-making Guide text file to practice using these features, or you can experiment with them in the text files for your own Guides.

Use a Word Processor to Modify Text Files

The text files we've included on the Starter Kit disk were created with SimpleText. Because these are text files, you can open any of them with a word processing program and then take advantage of typical word processing features—or tools—as you modify the file.

Tools

Several word processing tools should prove useful as you modify a text file. Most commercial programs offer some version of these tools, and you can probably find other features that will help you work more efficiently in your Guide's text file.

Search and Replace

The search feature alone makes using your word processor preferable to working with the file in SimpleText. You can quickly go to a specific word or phrase, hop from one section to the next, or determine whether a particular word or phrase is in the file.

If you need to change the terminology in your Guide during development, you can easily find and replace specific words or phrases with new versions. (You may want to use this feature even if you don't add or modify any content in the text file. You can open the text with the word processor, locate all instances of the words to change, and then copy those changes in the Guide Starter application for the Guide.)

Spell Checking

Spell checking is another tool you can use in the text file whether or not you make corrections or add content in that file. The spell checker will locate any misspelled words, which you can mark in the text file and then locate and correct in your Guide Starter application.

Page Numbers—Headers or Footers

When you use the text file as a printed document, whether for reference or as a working copy of your advanced features, you'll avoid confusion if you add page numbers to the document. You can also use a header or footer to record vital information about the file, such as a version number and revision date.

(If you simply want to use a printed copy of the text file for reference, you can print it in SimpleText, which adds page numbers automatically.)

Outlining

For extensive use of the text file, a word processor's outlining component can be quite effective. In many programs, for example, the outline view shows the first line of each paragraph, thereby giving you a quick way to analyze the structure of the Guide's content.

In addition, should you set up a template for working in the text file for a Guide, you can establish priority levels of headings and elements so that the outline view shows exactly the parts of the content you want to see.

Styles—Fonts, Formats, and Colors

Another word processing tool that may help when you are working with the text file is style sheets, or whatever automated formatting facility the program offers. By assigning different styles (and perhaps colors as well) to each element in the text file, you can easily see the content that you want to analyze or revise.

Mechanics

Some practical measures can help you avoid introducing errors in the content or having a failure at compile time.

Work in a Copy of the File

We've suggested this tactic before: Make a copy of the text file and modify it so that you keep the original untouched. You may need to consult the original version if you delete or modify part of the working copy and then later change your mind and want to reinstate the material.

Turn Off Special Marking Features

A number of word processing programs have special features that streamline operation by "marking" the text or converting some characters to another form, such as changing straight quotation marks to the curly form. To ensure that Guide Starter's compiler doesn't stumble over an unknown symbol in the text file, be sure to turn off the text-marking or conversion features while you are working with a Guide Script file.

The availability of these features varies somewhat by program, but typical features are

- a "fast save" option
- justified text margins
- conversion of straight quotation marks and apostrophes to the curly form, and
- bookmarks or other placeholders in the text.

If your program has similar features and you want to use them, do a test compile before you make many changes in the text file to be certain that those features won't affect the compile.

> **Tip** You can use keystrokes to introduce some special characters, including the dash, curly apostrophe, and curly quotation marks. Here's how:
>
> - For a dash, hold down the Option and Shift keys and press the hyphen.
> - For a curly apostrophe, hold down Option and Shift and press the close bracket ("]").
> - For curly quotation marks, hold down Option and press the open bracket ("[") for the opening mark, and hold down Option and Shift and press the open bracket for the closing mark.

Use Copy and Paste

As you work with the text file, take care to keep all Guide Script commands intact, including indentations and order of items. One way to minimize the risk of inadvertently changing Guide Script elements is to copy an entire panel within the text file, paste the copy where you want to add a new panel, and then change only the panel's text. (If you want the panel to have options such as a coach mark or Huh? button content, copy a panel that already has that option.)

Another way to create new content for a text file is to make your own template that contains copies of panels and other elements you want to use in a Guide. Use a copy of that template when you revise a Guide's text file, copying your new content from it as appropriate.

Develop a Routine and a Checklist

When you modify the text file, it's important that you make changes in all the necessary places. (See "Make All Necessary Changes or Additions" later in this chapter for guidelines for this procedure.)

To ensure that you don't overlook any of the secondary changes or additions, you could develop a set routine for working in the text file. For example, you might find it most efficient to make all changes in panels (Section V of the text file) as the first stage of your modifications. Then you could go to the sequences (Section VI) and revise any sequences that are affected by your panel modifications. Next, you might define any new coach marks that you added to panels (in Section IV). If you added custom prompts, you could note them next (in Section III). If you added new Topics or Index terms, you could add them next (in Sections VII and VIII). Finally, you could add your new content in the last four sections of the text file and then verify that the advanced features you added are referenced at other necessary places in the file.

Once you're comfortable with your routine for working in the text file, make a checklist to use each time you modify that file. The checklist could include each Guide element, along with the part of the file in which you must make corresponding changes if you change that element. The checklist should also reflect the order in which you proceed so that your work is methodical, thus further reducing the chances for error or omission.

Modify the Content of Text Files

Earlier in this chapter, we suggested that you can learn the fundamentals of scripting by working with the text file for your Guide. When you begin adding advanced features or new content to the text file, we recommend that you copy the parts of the file (or some other Guide's text file) you want to modify and then adapt them for your purposes.

Imitate and Borrow

Because the Guide Script syntax is exacting, you risk introducing errors if you create content by typing your own commands. Every angle bracket, comma, and quotation mark (straight, not curly) has a function. One misplaced or missing character can prevent the text file from being compiled successfully.

Make All Necessary Changes or Additions

As you copy parts of a text file and modify them to provide new content for your Guide, the following guidelines should help you determine where in the file to make changes:

- Every element in the text file must be defined. If you add a new element, you must define it in the appropriate section of the text file.
- Every panel, sequence, and coach mark must have a name. (The name can be a number.)
- If you delete an item, such as a coach mark, you needn't delete its definition. The opposite is not true, however. Deleting a definition and leaving the item will cause a compile failure.
- If you create a new panel, it must appear in a sequence. (It may appear in more than one sequence.)
- If you create a new sequence, it must have at least one panel, and it must appear in at least one of the lists in the Access window.
- If you add a new Topic in the Access window, it must have a corresponding sequence.

These guidelines also describe some of the content that Guide Starter creates for you.

In practice, these guidelines mean that you must change more than one section of the text file when you add or remove elements. Table 11-1 provides specific information about the parts of a text file to change for each type of modification you make.

Compile Often

If you're planning to make a lot of changes in the text file for your Guide, it's a good idea to complete part of the modifications and then compile. In this way, you'll smoke out any errors you've made in the new material before you've done all the work (and possibly repeated an error many times). Also, if there's a relatively small amount of new content, you'll have an easier time locating the errors.

Trial and Error

When you add advanced features for which you have few, if any, models, you may have to repeat the process of revising and compiling several times. Although this process can be tedious, you could discover new ways to use Apple Guide's features, and you're likely to encounter some surprises. Keep in mind that this is a new technology, and you're among its pioneers.

Table 11-1
Parts of the text file to change for Apple Guide elements

Apple Guide Element (Section Number)	Changes in Text File (Section Number)
Change text on existing panel (V)	No additional changes needed
Change text format on panel (V)	Rewrite text or copy from old format (V)
Activate Huh? button on panel (V)	Add Guide Script line to panel, with name of sequence assigned to the button (V)
Add custom prompt to panel (V)	1. Add prompt line (with prompt text) on panel (V) 2. Define prompt set (III)
Add coach mark to panel (V)	1. Add Guide Script line to panel, with coach mark number (V) 2. Define coach mark (IV)
Add new panel in a Topic (V)	1. Create all panel content (from copy of a similar panel, preferably) (V) 2. Add panel name or number to its sequence (VI) 3. If custom prompt or coach mark is used on panel, define prompt set (III) or coach mark (IV)
Delete a panel (V)	Delete panel number or name from its sequence (VI)
Move a panel to a new position (V)	Change its sequence to reflect the new position (VI)
Add a new Topic (VI)	1. Add Topic name under desired Header and Topic Area (VII) 2. Create panels for the new Topic (V) 3. If a custom prompt or coach mark is used on any panel, define prompt set (III) or coach mark (IV) 4. Create sequence for Topic (VI) 5. Assign Topic to one or more Index terms, if desired (VIII)
Delete a Topic (VII)	1. Delete Topic name (VII) 2. Delete Topic name from Index, if necessary (VIII) 3. Delete sequence for Topic (VI)
Add an Index term (VIII)	1. Add term to Index (VIII) 2. Assign one or more Topics (VIII)
Delete an Index term	1. Delete the term (VIII) 2. Delete all Topics and Headers assigned to the term (VIII)
Add custom logo in Access window (I)	1. Add <App Logo> command with path to the logo and name of the logo (I) 2. Prepare logo as a resource or a PICT

In some cases, you may not be able to get a feature to work even though you've used it in other circumstances. You may be working with a product that isn't compatible with Apple Guide, or that isn't scriptable (if you're using AppleScript scripts). If you have difficulty with a feature you think should work, look for examples among other Guide text files or consult other Guide authors if you can.

Guide Enhancements

Adding Panel Names

Guide Starter gives each panel a number rather than a verbal name. You may want to change those numbers to names if your Guide has many panels and you want a quick reference source that gives you a good idea of the panels' content.

Follow these steps to change the numbers to names in the text file with minimal risk of error or confusion:

1. In the text file (Section V), locate the <Define Panel> line for the first panel.
2. Click at the end of this line to the right of the four- or five-digit number.
3. Press the Tab key. Then type a name for the panel, with straight quotation marks immediately before and after it. (The name should describe the panel's content and be no longer than two or three short words.)

Figure 11-8 shows an example of the name.

4. Repeat steps 1–3 for each panel in the text file.
5. Print a copy of the text file so that you can easily see all the names you've added.
6. Copy the name you added for the first panel. Include the Tab space before the name.
7. Find the number of that panel in the sequences (Section VI) part of the text file. (You can either use the word processor's search capability to locate the panel number or consult your printed copy of the panels.)
8. Select the number of that panel in the sequence.
9. Paste the panel name, replacing the number.

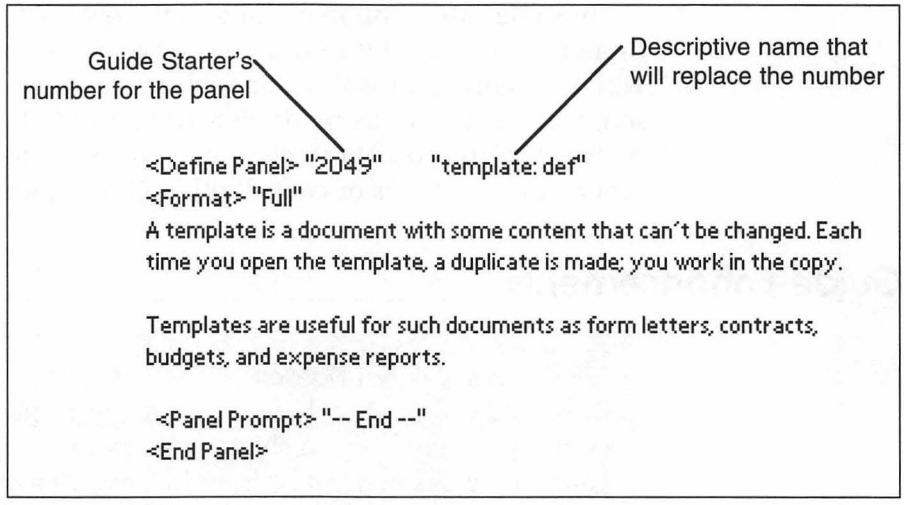


Figure 11-8
The number and replacement name for a panel

10. Repeat the search for this panel's number in case the panel appears in more than one sequence, replacing the panel's number with the name each time you locate that panel.

11. Continue replacing panel numbers with names in the sequences until you have changed all numbers to names.

12. Return to the panels (Section V) and delete the number for each panel, along with the Tab space that follows it. The panel name will then be in the correct location for both panels and sequences.

Changing the Huh? Button to a Summary Button

In most cases, you'll want to use the Huh? button in your Guide databases. Occasionally, however, you may want to replace the standard Huh? button with one that serves a different purpose, for example, to display a summary of all steps in a task, or to show a picture of an interface element that can't be coached. (A Guide database can have only one of this type of button. It is known as a "dimmable" button because you can make it active on a panel-by-panel basis.)

To use a different button, you must have the art for that button in a resource file. (You can use a resource editor, such as Apple's ResEdit, to create a resource file or to add a PICT graphic to the Standard Resources file that the compiler uses when building a database.)

The art for the Summary button in this example is in the Resource file named "Summary" on the Starter Kit disk. The text you need to add to the text file is included in the file named "Text file changes—Ch 11–14" on the Starter Kit disk.

1. Move the files "Template-making Guide text file," "Template Guide logo," and "Summary" to the Apple Guide Starter Kit folder, then open the text file. (Also open "Text file changes" if you want to copy from it.)

2. Choose Save As from the File menu. Rename the file "Template-making summary" and click Save.

3. In the text file (Section II), add <Resource> "Summary," 'PICT' after the two resource items.

4. Locate the first <Define Nav Button> line and replace it with the following line:

<Define Nav Button> "Summary",1015,1115,1125,DIMMABLE

5. Replace the last line of this section with the following line:

<Define Nav Button Set> "Start and Sum","GoStart","Summary"

6. In the panels section (V), locate panel 2313 and change the two lines above <End Panel> to read as follows:

(Click Summary below for a summary of steps to create a template.)
<Dimmable button data> "Summary", "Summary: Creating a template»1"

7. Add the following new panel at the end of the panels section:

<Define Panel> "6878"
<Format> "Full"
To create a template:
1. Create the document you want to save as a template.
2. Save the document in the program's regular format, with a name such as "Template Original."

3. Choose Save As from the File menu and:
• if you see a Stationery format, select it, change the document name, and click Save. (That's the last step.)
-- OR --
• if no Stationery format is listed, change the document name and click Save. (Go on to step 4.)
4. Select the icon for the document you just saved (with Save As) and choose Get Info from the File menu.
5. In the document's Info window, click to put an X in the box labeled "Stationery pad" at the lower right.
<Panel Prompt> "-- End --"
<End Panel>

8. In the sequences section (VI), replace the <Seq Nav Button Set> line with the following line:

<Seq Nav Button Set> "Start and Sum"

9. Add the following sequence at the end of the section:

<Define Sequence> "Summary: Creating a template»1","Summary: Creating a template"
<Seq Nav Button Set> "Start and Sum"
<Panel> "6878"
<End Sequence>

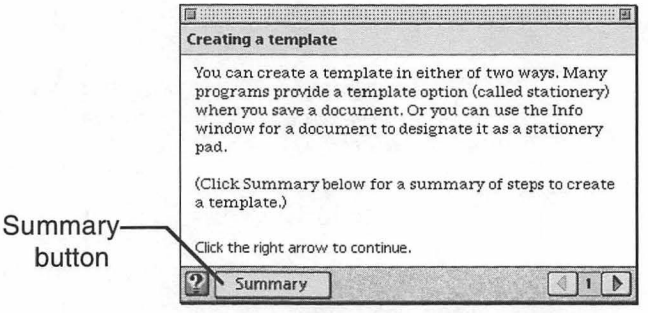


Figure 11-9
The Summary button replaces the Huh? button in this database

10. Save the text file.

11. Compile the text file by launching your Guide Starter application and choosing Build Database from Text File from the File menu.

12. When compiling is finished, put the Guide into the same folder as an application and then open that application.

13. Choose your Guide from the Guide menu and check the new content.

Figure 11-9 shows the Summary button on the panel where it is active.

At your leisure, look over the text file that contains your modifications for the Summary button. The changes you made replaced the Huh? button and added content that was assigned to the new Summary button. See if you can tell which new material is the added content and which replaces the Huh? button.

TWELVE
USING OTHER COACH MARKS

Apple Guide can not only display coach marks on menus and in windows. It also can coach most items on the screen, provided you give the right Guide Script commands and other necessary details about the computer and its software.

Types of Coach Marks

Apple Guide provides a variety of ways to define a coach mark. The type you use depends on the information you have about the computer's environment and the compatibility of the software you want to coach with Apple Guide.

Menu Coaches

As you know, Guide Starter automates menu coaches in the standard format of a red half-circle on the menu title and a red underlined menu item. You can change the look of a menu coach by specifying a different color or style for the menu item.

Examples of different menu coaches appear in the "Guide Enhancements" at the end of this chapter.

Window Coaches

Guide Starter also automates window coaches, using the default red circle coach mark. As with menu coaches, you can change the style of a window coach in the text file.

The "Guide Enhancements" at the end of this chapter include examples of window coach marks in different styles.

Dialog Item Coaches

One type of coach mark you must add in the text file is the item coach, which you generally use on part of a dialog box. To specify a dialog item coach, you need to know the number (called the "dialog ID") of the specific portion of the dialog box for the coach. You also usually need to know the four-character "signature" of the program that displays the dialog box.

Some examples of dialog item coaches are included in Guide Enhancements. See "Getting the Information You Need" later in this chapter for information about getting the dialog ID and program's signature.

AppleScript Coaches

Sometimes you need an alternative method of finding an icon or other item to coach, such as when you know its name but have no idea of its location. In most such cases, you can use an AppleScript script to find the item and report its location to Apple Guide for coaching.

Chapter 16 provides an introduction to AppleScript and offers examples of and details for specifying AppleScript coach marks.

Coach Mark Styles

Apple Guide offers several styles of coach marks in addition to the standard red circle and underlined red menu text. For example, you can choose from eight colors and eight text styles for menu items. You also can choose from several variations of marks so that your Guide's coach marks are as closely matched to the product and the instructions as possible.

Red Circle

The red circle coach mark is Apple Guide's signature, which was inspired by the legions of chalkboards, coaches, and commentators in our sports-saturated society. As a guide to the Macintosh, this dynamic orb works beautifully—it catches the user's eye and quickly eliminates the confusion that frequently goes along with an unfamiliar task.

For most cases, the red circle is an appropriate indicator. But a bold mark such as this sometimes obscures part of the interface the user needs to see or simply takes up too much room on the screen. Or sometimes the circle may not be precise enough to isolate, say, only one item in a list.

Red Underline

One alternative coach mark is the red underline. This mark is particularly useful for indicating a spreadsheet cell or a text-entry area in a complicated information form. When you specify a red underline, the coach mark is approximately the length of the widest part of the red circle coach.

An example of the red underline coach mark appears in "Guide Enhancements" at the end of the chapter.

Green X

The green X is another coach mark that's best used in special situations. Because this mark is compact, for example, it can be used on the PowerBook's small screen without obscuring too much content.

See the "Guide Enhancements" section for a sample of this coach mark.

Red Arrow

The red arrow is somewhat more versatile than the other coach marks. You describe this arrow with two numbers that indicate its orientation on the screen. You can use this mark to show the direction in which a user would move something, such as dragging a disk's icon to the Trash. (Getting the location and direction of the arrow just right takes some experimentation, however.)

The "Guide Enhancements" section includes a sample of the red arrow coach mark.

Getting the Information You Need

Depending on the type of coach mark you want to use and the product your Guide covers, you may need to include one or two items of technical information in the definition of the coach mark. Although these details will be unfamiliar to those of us who aren't programmers, they are relatively easy to obtain.

Dialog Item IDs

When you specify a coach mark for an item in a dialog box, you must include the dialog ID number for the item to be coached. You can get ID numbers with a resource editor such as Apple's ResEdit. This program is included with several programming-related books, including *ResEdit Complete*, Second Edition, by Peter Alley and Carolyn Strange (Addison-Wesley, 1993) and *The ResEdit All Night Diner*, by Dave Ciskowski (Hayden Books, 1993).

If you're not familiar with resource editors, consult one of these books or an experienced Macintosh programmer for help and always work with a copy of the program for which you need information. To find the dialog IDs, launch ResEdit and use it to open a copy of the program you want. Then use the DITL item for the program in order to show the dialog boxes; do this by double-clicking DITL and then double-clicking the items listed in the DITL window. If you want dia-

If you don't have ResEdit or another resource editor, you can make a logical guess as to the number of a dialog item and then change the number later if necessary. If the programmer followed Apple's guidelines, the button with a bold border is number 1. Usually the other items in a dialog box are numbered in a somewhat logical way, with the most standard items having lower numbers (see Figure 12-1 for examples). Count the items—the places where the user could indicate a response—and give the item you want to coach a number that seems logical. (Note that the dialog box in Figure 12-1 has some numbered items that aren't for user input, so guessing correctly may be a challenge for a big dialog box such as that one.)

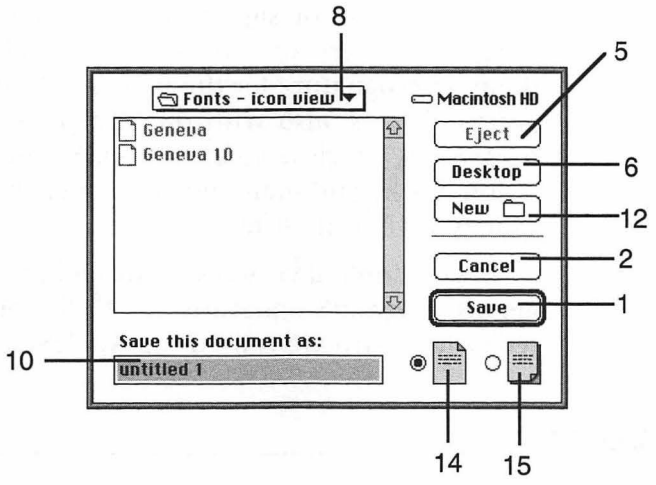


Figure 12-1
The dialog ID numbers for the Save dialog box in SimpleText

log IDs for a large dialog box, such as the Save dialog, use the sizes in the list to determine the most likely item. In the resource editor, choose Show Item Numbers from the DITL menu. (Don't worry if these instructions sound like Greek; the procedure is a lot less obscure when you're using ResEdit.)

Figure 12-1 shows the dialog ID numbers for items in the Save dialog box for the SimpleText program that's included with System 7.5.

Application Signatures

Another essential piece of information for many coach marks is the application's signature, the four-character code that uniquely identifies that program. You can get the signature easily with ResEdit by doing the following:

1. Launch ResEdit without opening any application.
2. Choose Get File/Folder Info from the File menu.
3. Select the program in the directory dialog box and click Get Info.

   The item you want from the Info window is the creator (near the top center of the window).

The creator, or signature, for SimpleText is 'ttxt'—note the all lowercase letters and the straight, single quotation marks around them. The signature for the Finder—'MACS'—is expressed in all uppercase letters, also with the straight single quotation marks. You must always enclose an application's signature within the single quotation marks and maintain the uppercase and lowercase characters exactly as they are listed.

If you don't have access to ResEdit, you can use an AppleScript to get the program's signature. See "Defining Context Checks" in Chapter 17 for instructions on using AppleScript for this purpose.

Special Cases

There are always exceptions to any rule, and coach marks have some of their own. Two Macintosh menus—the Guide menu and the Application menu—don't respond to conventional coach mark definitions. You can use a window coach to circle the menu title for each of these menus, but you can't coach the items listed in either menu. (The ¬ symbols below are line continuation symbols to let long lines fit within the margins of our page.)

The Guide Script command to identify the Application menu is

```
<DEFINE WINDOW COACH> ¬
"CoachMark: App menu", 'MACS', REDCIRCLE, DESKTOP, RECT(5,-35,15,-16), TOPRIGHT
```

The Guide menu coach mark command is

```
<DEFINE WINDOW COACH> ¬
"CoachMark: 60", 'MACS', REDCIRCLE, DESKTOP, RECT(5,-61,15,-41), TOPRIGHT
```

By way of explanation:

- 'MACS' is the Finder's signature, so the Finder is the target application for this coach mark.
- REDCIRCLE is the standard coach type.
- DESKTOP is the "window" (in place of the usual FRONTWINDOW as the target window for the coach).
- RECT(5,-61,15,-41) are the coordinates (top, left, bottom, right) for a rectangle that forms the boundaries of the coach mark.
- TOPRIGHT is the corner of the screen from which the location of the coach's rectangle is measured.

The last item, TOPRIGHT, can be very useful for locating a coach mark when you don't know the size of the user's screen. Apple Guide uses TOPLEFT—the top left-hand corner—as its default if no other parameter is given as the last item in a window coach definition. The parameters for the bottom corners are, not surprisingly, BOTTOMLEFT and BOTTOMRIGHT.

You can add these commands to the text file for your Guide whenever you want to coach the Application and Guide menus.

Guide Enhancements

Coaching Items in a Dialog Box

As part of creating a template, as discussed in Chapter 11, here's how you would define coach marks on the Save dialog box in the SimpleText program.

1. Open the original "Template-making Guide text file" and save it as "Template-making #1 coach."

2. Go to the coach marks section (IV) and add the following three lines. (To get the "»" character, hold down the Option and Shift keys and press the backslash key, which is immediately below the Delete key.)

```
<Define Item Coach>
"CoachMark»3",'ttxt',REDCIRCLE,FRONTWINDOW,DialogID(15)
<Define Item Coach>
"CoachMark»4",'ttxt',REDCIRCLE,FRONTWINDOW,DialogID(10)
<Define Item Coach>
"CoachMark»5",'ttxt',REDCIRCLE,FRONTWINDOW,DialogID(1)
```

3. Go to the panels section (V) and revise or add the following panels (so that they include the three new coach marks).

Revise panel 3650 to read as follows:

```
# Revised for coaches
<Define Panel> "3650"
<Format> "Body"
Using the SimpleText program on your hard disk, create the document
you want to use as a template. Include all content that should appear in
every copy.
```

When the template content is complete, save the document with a name such as "Template Original." (This is the version you can use to revise the template later.)

<Format> "Tag"
Do This
<End Panel>

Revise panel 5582 to read as follows:

#revised for coaches
<Define Panel> "5582"
<Format> "Body"
<Coach Mark> "CoachMark»3"
<Format> "Body"

In the dialog box, click the stationery radio button (on the lower right, with the icon that resembles a notepad) to activate it.

<Format> "Tag"
Do This
<End Panel>

4. Add the following two new panels immediately after panel 5582:

#New for item coaches - number 1
<Define Panel> "7145"
<Coach Mark> "CoachMark»4"
<Format> "Body"
Type a name for the template.

<Format> "Tag"
Do This
<End Panel>

#New for item coaches - number 2
<Define Panel> "7236"
<Coach Mark> "CoachMark»5"
<Format> "Body"
Click Save.

The next time you open this document, the program will make a copy and you'll have to name it before you can work with it.

<Format> "Tag"
Do This
<End Panel>

5. Delete panels 5745, 6361, and 6470.

6. Change the sequences (Section VI) to reflect your panel changes. First, insert the two new panels (7145 and 7236) immediately below panel 5582 in the sequence "Creating a template»1."

Then delete the last three panels in the sequence (5745, 6361, and 6470).

7. Save the revised text file with a new name.

8. Compile the revised text file by launching your Guide Starter application and choosing Build Database from File from the File menu.

Changing the Style and Color of a Menu Item Coach

At your leisure, use one of the definitions to vary the look of menu item coach marks:

➤ For different text style in the item coached:

<Define Menu Coach> "CoachMark»1",FRONT,REDCIRCLE,"File",
"Save As...",RED,BOLD
<Define Menu Coach> "CoachMark»2",FRONT,REDCIRCLE,"File",
"Get Info",RED,OUTLINE

Your options for the text style (the final word in the definition) include PLAIN, BOLD, CONDENSE, EXTEND, ITALIC, OUTLINE, SHADOW, and UNDERLINE.

➤ For different text color in the item coached:

<Define Menu Coach> "CoachMark»1",FRONT,REDCIRCLE,"File",
"Save As...",CYAN,BOLD
<Define Menu Coach> "CoachMark»2",FRONT,REDCIRCLE,"File",
"Get Info",CYAN,OUTLINE

Your options for the text style (the final word in the definition) include BLACK, BLUE, CYAN, GREEN, MAGENTA, RED, WHITE, and YELLOW.

Changing the Style of a Dialog Item Coach

Use one of the following definitions to create a different coach mark style in a dialog box:

➤ To display a red line under the dialog box item

```
<Define Item Coach>
"CoachMark»4",'ttxt',REDUNDERLINE,FRONTWINDOW,DialogID(10)
```

➤ To put a green X in the dialog box item

```
<Define Item Coach>
"CoachMark»5",'ttxt',GREENX,FRONTWINDOW,DialogID(1)
```

Try out these changes by substituting them for the corresponding coach marks in your revised copy of the Template-making Guide text file.

Using Alternative Coach Marks on the Desktop

Try out one of the following coach marks for still more variations of these handy on-screen indicators:

➤ To display a green X on the Desktop, near the upper right-hand corner of the screen

```
<DEFINE WINDOW COACH> "CoachMark: desktop 1", 'MACS',¬
GREENX, DESKTOP, RECT(30,-85,50,-65), TOPRIGHT
```

➤ To display a red arrow that points toward the hard disk icon on the desktop

```
<DEFINE WINDOW COACH> "CoachMark: desktop 2", 'MACS', ¬
RedArrow(7,3), DESKTOP, RECT(50,-35,90,-10), TOPRIGHT
```

By way of explanation:

➤ 'MACS' is the Finder's signature, so the Finder is the target application for this coach mark

➤ RedArrow(7,3) is the coach type. The number in parentheses orients the arrow on the screen from a rectangular pattern of numbers that looks like this:

```
1   2   3
8       4
7   6   5
```

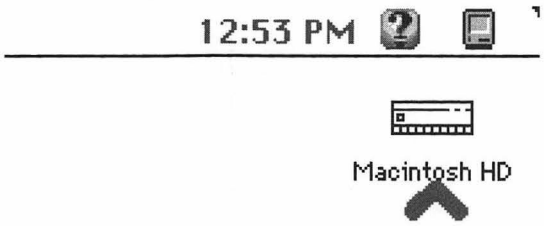


Figure 12-2
The red arrow coach mark on the desktop

➤ DESKTOP is the "window" (in place of the usual FRONT-WINDOW as the target window for the coach).

➤ RECT(50,-35,90,-10) is the coordinates (top, left, bottom, right) for a rectangle that forms the boundaries of the coach mark.

➤ TOPRIGHT is the corner of the screen from which the location of the coach's rectangle is measured.

Figure 12-2 shows the arrow coach mark defined in the previous example.

Try out all the coach marks and their variations in your Guides to determine which ones work best for your instructional objectives.

THIRTEEN
ADDING GRAPHICS AND QUICKTIME MOVIES

Because Apple Guide's coach marks show users where to take actions on the computer's screen, Guide databases usually don't have to rely on graphics or animations to provide instructions. Occasionally, though, a product doesn't cooperate with the coach mark technology. A picture may be needed.

In addition, graphics or QuickTime movies can add useful information to a Guide. For example, a database that demonstrates new features of commercial software could benefit from a QuickTime movie. Instructions for a new telephone system in an office setting could include a labeled picture to orient the staff to the new procedures. A business that hires temporary workers to process information could integrate basic instructions for those tasks with pictures of seating plans, maps to meeting rooms, and organizational charts to help the newcomers.

Pictures or Movies for Instruction

Certainly pictures and movies have a place in instruction. Many procedures benefit from the sort of active demonstration that QuickTime movies

provide, such as how to hold and move the mouse (for first-time users) or how to maintain ergonomically correct hand and wrist positions at the computer (for everyone).

Refrain from Using Interface Pictures

Traditional graphics can also add value to a Guide. However, mixing graphics and coach marks to show an interface is usually not advisable. As we've said previously, users quickly develop expectations when working with software. They could become confused or frustrated if some instructions have coach marks, while others have pictures for reference and no coach marks. More important, numerous user studies, including some with Apple Guide, have shown that users fairly often mistake a picture of the interface for the real interface, and they can become really confused if they click on a graphic and get no results.

If coach marks don't work with a program for which you're preparing a Guide, you may still decide that users will be better served by pictures of the interface than merely with written descriptions. If you do use pictures, you can reduce the chances of users' mistakenly trying to manipulate those pictures by using any of the following strategies:

- Label the picture as an example, either on the picture itself (use a simulated rubber stamp across it, for example) or in the prompt on that panel.
- Reproduce the interface element at a noticeably different scale from the actual interface.
- Show the relevant part of the interface in regular color or grayscale, but dim other areas of the picture so that it is obviously not the actual interface.
- Put labels on the picture (useful or necessary for instructional purposes, anyway) to distinguish it from the interface.
- "Hide" the picture in a panel attached to the Huh? button so that only users who need its extra help will see it.

For the most part, QuickTime movies don't have this problem, although users may not be familiar with the standard QuickTime player and window. It's always wise to tell users how to start and stop a movie. Don't assume they know.

Consider Space and Time

Depending on the method you'll use to distribute your Guide and the size of the database, you may need to limit the amount of graphics related to a Guide. QuickTime movies and color PICT files (the type of graphics Apple Guide accepts) can be very large, and movies must be provided with the Guide for its users. If you plan to distribute a Guide on floppy disks or post it on an information service, extra graphics files may increase your costs sharply, even if you compress the files for distribution.

Similarly, users generally have a variety of Macintosh models. Some of these are rather slow at processing large graphics files or QuickTime movies. Be sure to weigh the advantages of each graphic or movie against the potential for users' impatience or frustration as they wait an extra second or two for the pictures.

As you consider the advantages and disadvantages of adding graphics, also keep in mind that the objective of a Guide usually is to help users get on with their own work. If the visual images serve that purpose and are necessary for adequate explanation, you should probably use them.

Pictures or Movies for Reference and Demonstration

For Guide databases that contain reference information or a mixture of reference and instructions, graphics can provide a shortcut. A picture of a fax machine or scanner with succinct labels and possibly numbers indicating steps could be a handy reminder for people who have a basic understanding of the procedure but haven't retained the details.

A movie can show off features of hardware or software, and a Guide database could provide an excellent marketing tool for this purpose. Because the Apple Guide window is interactive and always available (if the Guide is stored with the programs it covers), it could be the basis for an unattended demonstration of a product.

As you become experienced at creating Guide databases, you may find that working in Guide Starter and later modifying the text file is a good way to build prototypes for demonstrations or presentations.

Add Visuals to Panels

Apple Guide has built-in commands for including graphics and QuickTime movies in a database. The procedure is simple, although you'll probably want to do some experimenting to get the visual elements placed where you want them. Among the requirements for including graphics and movies are the following:

- Graphics or movies can be placed on a panel.
- Graphics files must be in the PICT format or stored as resources (which you create with a resource editor such as ResEdit).
- If the graphic is a PICT file, that file must be in the same folder as the text file when you compile the Guide.
- Graphics cannot be larger than the maximum height or standard width of the Presentation window (see "Guide Enhancements," next section).
- QuickTime movies must be files in the QuickTime format, and they must remain in the same folder as the Guide (or they won't be available to users).

Guide Enhancements

Adding a Graphic by Means of an Extra Navigation Button

When a graphic is intended for reference or as a secondary part of an instruction, you can keep it available to users throughout a sequence (or all sequences in a Guide) by attaching it to an added navigation button. This new button appears to the right of the Huh? button, but unlike Huh?, the new button is always active. (A Guide can have only one "dimmable" button, which is the Huh? button.)

The following example makes these changes in a text file to accommodate the graphic:

- changes the maximum size of the Presentation window
- adds a new button
- adds art for the new button in a resource file
- adds the graphic in a panel.

The text file and the button art graphic for this example, "Old Salt Inventory text file," "Flags," and "Flags.PICT," are included on the Starter Kit disk. Only the sections related to the features listed above are presented here. You can work with the text file provided or add the content that follows to your own Guide's text file.

Follow these steps, to change the Presentation window size and add button art, a second button, and a graphic to a Guide's text file:

1. Move the files "Old Salt logo," "Flags," "Flags.PICT," and "Old Salt Inventory text file" to the Apple Guide Starter Kit folder.

2. Open the original "Template-making Guide text file" and save it as "Old Salt Inventory."

3. In the "Old Salt Inventory text file," go to the Guide Startup Info Section (I) and add the following between <Howdy> and <Version> (near the end of the section):

```
# set maximum height of presentation panel at 500 pixels
<Max Height> 500
```

By way of explanation:

- A maximum of 500 pixels is larger than some screens may be able to display, but if you have a large graphic you can begin with that. If your graphic is too large for a 500-pixel window, you'll know you must reduce it. If the graphic is smaller than 500 pixels, Apple Guide makes its window only large enough to show the graphic, so the window won't automatically be 500 pixels high.
- You can't adjust the width of the Presentation window. Its text area (and area for graphics) is approximately 320 pixels wide.

4. Go to Section II and add <Resource> "Flags," 'PICT' below the two resource items.

5. Add the following lines after the last <Define Nav Button> and before the first <Define Event>:

```
<Define Nav Button>    "Flags",1660,1670,1680,¬
LaunchNewSequenceNewWindow("Nautical flags - alphabet»1")
```

By way of explanation:

➤ <Define Nav Button> is the command that introduces this element into the database.

➤ "Flags" is the name of the new button (and the name of a resource file with art for the button).

➤ 1660, 1670, and 1680 are resource numbers of art for the new button in the resources file "Flags" (located on the Starter Kit disk).

➤ LaunchNewSequenceNewWindow directs Apple Guide to launch a sequence in a new window (as the Huh? button does) when the new button is clicked.

➤ The sequence to be launched in the new window is named "Nautical flags - alphabet»1."

Figure 13-1 shows the new Flags button in the Presentation window.

6. Add the following lines as the last entry in Section II:

```
<Define Nav Button Set> "Start and Huh and ¬
Flags","GoStart","Huh?","Flags"
```

By way of explanation:

➤ <Define Nav Button Set> is the command that introduces the new "Flags" button as part of a set that can be used with any sequence.

7. Add the following panels in Section V:

```
#now define the real panels.
<Define Panel> "flags - nautical"
<Format> "Full"
Nautical flags now come in only one size: 18 by 24 inches. These flags
are mainly used for decoration, but some long-time customers continue
to purchase full alphabetical sets.

The Caspar Pond Association has a yearly contest with these flags. Be
sure to have at least two of each on hand before June 15th for this event.
  <Panel Prompt> NONE
<End Panel>
```

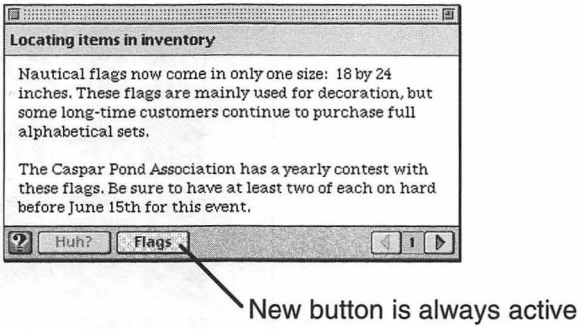


Figure 13-1
Flags button makes a graphic available from all panels in a sequence

<Define Panel> "flag orders 1"
<Format> "Full"
You locate nautical flags in the warehouse by eye. A chart of all flags is posted at each end of the flag aisle, and a small version of each flag is posted below its location on the shelf.

(Click Flags below to see a picture of the nautical flags. Click Huh? for background info.)

<Dimmable button data> "Huh?", "Nautical flags»1"
<Panel Prompt> NONE
<End Panel>

<Define Panel> "Nautical flags - alphabet"
<Format> "Full"
<PICT> "Flags.PICT",CENTER,"Flags.PICT"
<Panel Prompt> NONE
<End Panel>

By way of explanation:

- The first two panels are standard content that you've seen before.
- The third panel, "Nautical flags - alphabet," contains the graphic, named "Flags.PICT."
- <PICT> specifies the graphic by name.
- CENTER tells Apple Guide to center the graphic in the panel's text area.

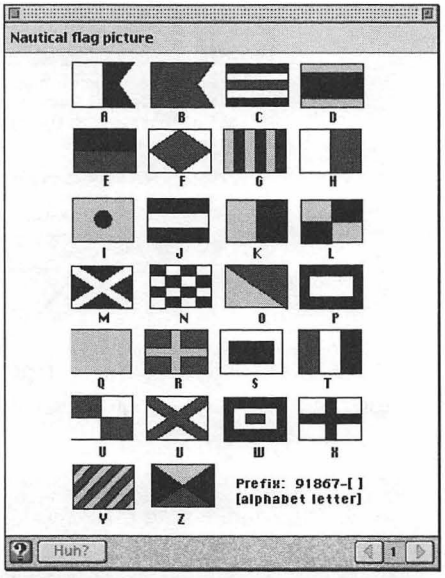


Figure 13-2
A graphic centered in the Presentation window

➤ "Flags.PICT" is the name of the file to be displayed on black-and-white monitors (in this case, the same file). This parameter of the <PICT> command is optional. (You can prepare a black-and-white graphic if you know that some users do not have color or gray-scale monitors.)

Figure 13-2 shows the graphic in the Presentation window.

8. Go to the sequences section (VI) and add the following content:

```
#now define the real sequences.
<Define Sequence> "Nautical flags»1","Nautical flags"
   <Seq Nav Button Set>"Start and Huh and Flags"
  <Panel> "flags - nautical"
<End Sequence>

<Define Sequence> "Locating items in inventory»1","Locating items in
inventory"
   <Seq Nav Button Set>"Start and Huh and Flags"
  <Panel> "flags - nautical"
  <Panel>  "flag orders 1"
<End Sequence>
```

```
<Define Sequence> "Nautical flags-alphabet»1", "Nautical flags-alphabet"
  <Seq Nav Button Set>"Start and Huh"
 <Panel> "Nautical flags - alphabet"
<End Sequence>
```

By way of explanation:

- The first two sequences correspond to the standard panels shown above and are also standard, with one addition: The <Seq Nav Button Set> listed on each of those panels includes the new button "Flags."

  (Look at other text files and you'll notice that every sequence has the <Seq Nav Button Set> command.)

- The third sequence, "Nautical flags - alphabet" is the one that opens when a user clicks the "Flags" button. Its one panel contains the graphic.

9. Go to the Topic Areas and Topics section (VII) and add the following content:

```
<Header> "Instructions for"
  <Topic> "Locating items in inventory","Locating items in
inventory»1"
  <Topic> "Re-ordering nautical flags","Default"
```

By way of explanation:

- The Topics listings don't require any special content for the new button or the graphic.
- Some Topics are placeholders for which content hasn't yet been written, so Guide Starter has given them its "Default" sequence to prevent compiling errors.

10. Go to the PICTS and Movies section (XI) and add comments, preceded by the number sign (#), noting that you've added a graphic and a new button.

11. Make sure the logo and two "Flags" files are in the same folder as Guide Starter. Then compile the text file.

Adding a QuickTime Movie

You add a QuickTime movie to a panel in much the same way as you add a graphic. In this example, there aren't any added buttons or changes in window size, just the movie on a panel.

Follow these steps to add a QuickTime movie to a Guide's text file. (The relevant files are provided on the Starter Kit disk.)

1. Move the files "Naomi," "Naomi.PICT," and "QuickTime Movie text file" to the Apple Guide Starter Kit folder. Open the text file.

2. In the QuickTime Movie text file, go to the panels section (V) and add the following panel:

```
<Define Panel> "QT movie"
<Format> "Full"
                Double-click the picture to see and hear Naomi's
                  message for her sister.
<QuickTime> "Naomi",CENTER,PLAIN, "Naomi.PICT"
  <Panel Prompt> NONE
<End Panel>
```

By way of explanation:

- The text is indented so that it's centered above the movie on the panel. Getting this centered takes experimentation (guessing, compiling, adjusting, and so on).
- <QuickTime> is the command that associates a movie with the panel.
- "Naomi" is the name of the movie.
- CENTER directs Apple Guide to center the movie on the panel's text area.
- PLAIN tells Apple Guide to display the movie without the QuickTime window around it.

Figure 13-3
A QuickTime movie centered on a panel

You can use CONTROL instead of PLAIN if you want the QuickTime control window displayed with the movie. Or use BADGE to display a badge on the movie—when it's not playing—that the user double-clicks to show the control window. Try these three alternatives to see which you prefer.

➤ "Naomi.PICT" is a graphic that appears on the panel if the movie is not in the folder with the Guide. (The graphic must be the same size as the movie.)

Note: Apple Guide does not store QuickTime movies as part of the Guide. You must be sure that any movies used in a Guide are provided to users and that they are kept in the same folder as the Guide.

Figure 13-3 shows a QuickTime movie on a panel.

Experiment with the examples in this chapter and the files provided on the Starter Kit disk. You'll very likely develop some good ways to use graphics and movies in your Guides.

FOURTEEN

BUILDING TOPICS WITH BRANCHES AND DECISION PANELS

Many tasks naturally divide into separate operations or variations of the same operation. You can reduce the number of tasks in a Guide, and thus make the user's job of finding tasks easier, by building branches into many of the tasks.

Another way branching tasks streamline a user's access to instructions is by concentrating all the information about a Topic—printing, for example—in one set of panels. The user can choose one branch, say for using a network printer, read its instructions, then move back through the panels to the decision point and choose another branch, for example, for a printer connected directly to the computer. No time is lost going back to the Access window, scrolling through lists of Topics, and hunting for the right task.

Plan Radio Buttons and Branches

When planning your Guide's content, you already consider how to divide general subjects into manageable and logical tasks. You can use the added flexibility of building tasks that branch to group similar operations together in whatever arrangements will best assist the users of your Guide.

For example, if the Guide's users are Macintosh veterans who already know the kinds of tasks the computer can automate for them, you can probably arrange the database into a small number of "supertasks." Each of these large Topics could combine several related tasks as branches.

If the Guide's users are less familiar with the computer and its uses, you might want to keep most tasks small so that users can get an idea of their options as they browse the Access window's lists. Of course, the number of tasks and size of the database are also considerations as you determine how much of the content can be subdivided into branching tasks.

Yet another great benefit of building branches into tasks is that you can deliver the correct instructions to users even when you don't know which version of a product they have. By writing a branch for each possibility, you can design the task so that users specify which branch they want to use. Or you could include a context check that determines the user's system capabilities and displays the appropriate branch.

Likely Candidates for Branching

You can present many kinds of information in a branching arrangement. The types of tasks that lend themselves to this treatment include the following:

- how to use alternative equipment, such as a network printer or a direct-connect printer
- how to do a task that can be performed several different ways, such as creating a template file
- how to do standard operations with different software, such as printing with the standard system software or printing with QuickDraw GX (a sophisticated type-handling and printing software that is included but not installed automatically with System 7.5)
- different levels of instruction—a short branch with a brief summary of steps and another with multiple panels and extensive context checks and coaches
- a monitored tutorial, which tracks the user's progress and delivers one branch of information or another depending on how successful the user has been up to that point.

Branches

When you have a general idea of how extensively you want to use branching tasks, study the list of Topics you've developed (or are devising) to assess whether your expectation fits the Guide's content. Then make a tentative list of the branching tasks and sketch each one on paper.

Ideally, all the branches of a task should be roughly equal in length and complexity. If you map the content of a task and find that one branch has five panels and another has a dozen or more steps, go over the content to see if there's a better way to subdivide the information.

You can use branches within branches if necessary, but such intricate structures should be kept to a minimum. Remember that users easily get lost or impatient when they don't have an idea where they're going (or where they've been).

Decision Points

For most branching tasks, you can put the decision point, a panel with radio buttons, at or near the beginning. As you plan the flow of information for a Topic that subdivides, consider what part of the information applies to all users. The logical and efficient place to put a decision panel is at the end of the information that's common to everyone who chooses the Topic.

Similarly, many Topics contain information that all users should see after they've gone through one of the task's branches. For these situations, you can design your sequences so that they merge at the end.

Write Content for a Branching Task

We can give you a better idea of how to create branching tasks by showing you a familiar one. The rest of this chapter shows you how to modify the text file for building a template so that it includes branches for the two ways to accomplish the task.

The original text file for this Guide, Template-making Guide text file, is on the Starter Kit disk. If you tried out the revisions detailed in Chapters 11 and 12, you should also have one or two revised versions of this file. (Use "Text file—to add" for changes here.)

Radio Button Panels

As you look at the original version of the text file for the Template-making Guide, you'll notice that the first panel (number 2313) of the task "Instructions for creating a template" notes that you can create a template in either of two ways. This looks like a good place to give users the choice of which method they'd like to use.

You can do that by adding two radio buttons to this panel. The revised version would look like this:

```
# Revised for radio buttons
<Define Panel> "2313"
<Format> "Full"
You can create a template in either of two ways. Many programs provide
a template option (called stationery) when you save a document. Or you
can use the Info window for a document to designate it as a stationery
pad.

Which do you want to do? Click one:
 <RADIO BUTTON>  "save a document as stationery using SimpleText",
true,,, ,APPLEGUIDE
 <RADIO BUTTON>  "use the Info window to designate a stationery
pad", false,,, ,APPLEGUIDE

(Click Huh? below for a definition of "template.")

  <Dimmable Button Data> "Huh?","Template»1"
 <Panel Prompt> "Make your choice, then click the right arrow."
<End Panel>
```

The new items on the panel are

- the comment (first line), which is very helpful for keeping track of your revisions,
- an added line of text that asks users to make a choice,
- the radio button descriptions, and
- a prompt line.

The radio buttons are the unfamiliar element, so we'll discuss those lines in some detail.

Figure 14-1 shows the two radio button statements with labels that describe each element.

Unlike some other types of buttons, you don't have to provide art for a radio button. Apple Guide puts the button on a panel that contains a radio button statement.

True or False

As the statements in Figure 14-1 show, radio buttons can have a value of true or false. One button must be true by default, and only one button can be true at a time. When the user clicks a radio button, its state changes.

The state of the panel's radio buttons determines which branch of the task a user sees. The Guide Script statements in the sequence (Section VI of the text file) for the task contain directions for branching that respond to the true or false condition of the radio buttons.

Location on Panel

The placement of radio button statements within the panel text determines the buttons' location. Note that the radio button statements on the revised panel 2313 appear before the parenthetical sentence referring users to the Huh? button. That placement is intentional because

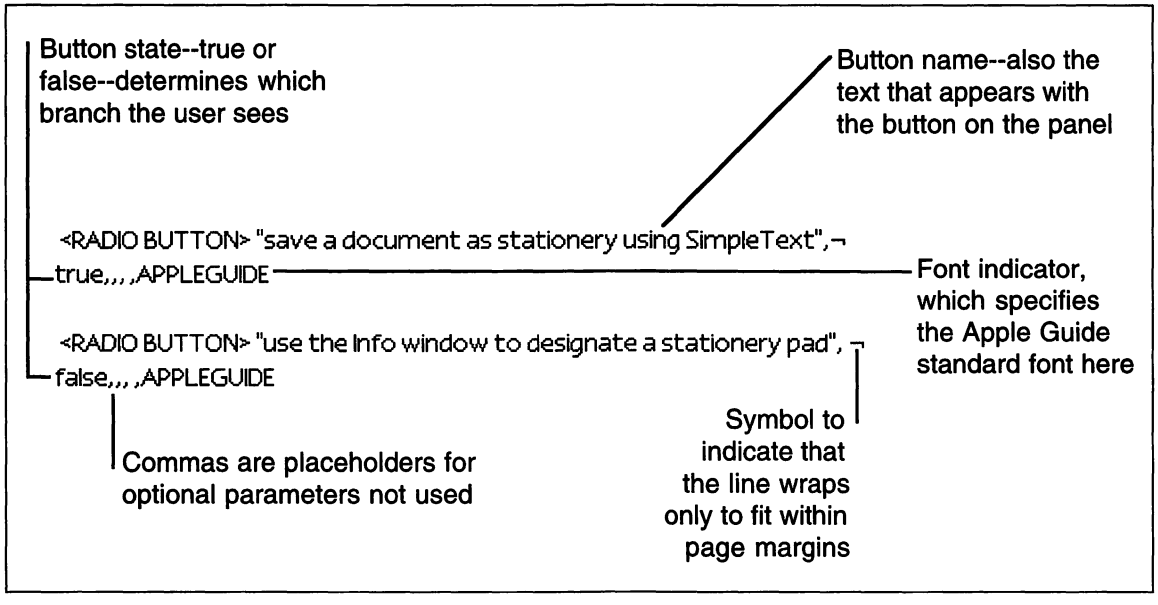


Figure 14-1
Radio button statements for a decision panel

the user's priority should be choosing a radio button (and the mention of the Huh? button benefits from being the closest text to that button).

Font

You can also specify a font for the radio button titles, which appear with the button on a panel. The APPLEGUIDE designation in Figure 14-1 specifies that the standard Apple Guide font (10-point Espy) be used. The other choice available is the system font used by the current script system on the Macintosh. For the Roman script system, which is used in the United States, the font is 12-point Chicago. The system font is the default radio button font.

Sequences for Branches

The sequences for branching tasks can be handled simply with the "If, then, else" logic that is frequently used for programming, spreadsheet formulas, and other data-handling operations. For our example of the template, the limit of only two choices also makes for uncluttered scripting.

Here's the revised sequence for "Creating a template," which now contains two branches:

```
# Revised for radio button task
<Define Sequence> "Creating a template»1","Creating a template"
   <Seq Nav Button Set> "Start and Huh"
  <Panel> "2313"     #radio buttons added
  <If> radioButtonState("save a document as stationery using Simple-
Text","2313")
     <Panel> "3650"
     <Panel> "5297"
     <Panel> "5582"
     <Panel> "7145"
     <Panel> "7236"
  <Else>
     <Panel> "7912"
     <Panel> "5297"
     <Panel> "7965"
     <Panel> "5745"
     <Panel> "6361"
     <Panel> "6470"
     <Panel> "7518"
  <End If>
 <End Sequence>
```

Notice that this long sequence contains <If> (line 6) and <Else> (line 13) statements. The <If> line gives the radio button state, the name of one button, and the name of the panel on which that button appears. The "true" condition of this radio button is assumed, as is the "then" part of "If, then, else" logic. In essence, the <If> statement tells Apple Guide that if the radio button labeled "Save a document as stationery using SimpleText" is active, then go to panel 3650.

The <Else> statement covers the exception: If the "Save a document..." button is not active, Apple Guide is directed to panel 7912. In this sequence, there are no panels after the last one in each branch. If the task had any such panels, they would appear after <End If> and before <End Sequence>.

Now you have the two central elements in this revision to create branches for the template-making task. There are several other changes to make in the text file before you can compile it, however. These changes are presented in the following "Guide Enhancements."

If all this scripting lingo has gone by too quickly, think through the branches you'd make in the template task, or revise content from your own Guide to include branches. You'll quickly grow comfortable with this approach to delivering information.

Guide Enhancements

Revising the Template-making Guide Text File for Branches

Here are the complete instructions and content for creating a branching version of the template-making task. It uses all panels from the two earlier versions of this task (Chapters 11 and 12), plus some new and revised content.

1. Open the original version of the Template-making Guide text file. In the prompts section (II), add the following prompt set:

<Define Prompt Set> "Make your choice, then click the right arrow.","Make your choice, then click the right arrow.","Make your choice, then click the right arrow.","Make your choice, then click the right arrow.","Make your choice, then click the right arrow."

2. Go to the panels section (V) to revise and add content. First, revise panel 2313 to add radio buttons, a prompt, and some new text, as noted next (you've seen this panel previously):

```
# Revised for radio buttons
<Define Panel> "2313"
<Format> "Full"
You can create a template in either of two ways. Many programs provide a template option (called stationery) when you save a document. Or you can use the Info window for a document to designate it as a stationery pad.

Which do you want to do? Click one:

 <RADIO BUTTON>   "save a document as stationery using SimpleText", true,,, ,APPLEGUIDE
 <RADIO BUTTON>   "use the Info window to designate a stationery pad", false,,, ,APPLEGUIDE

(Click Huh? below for a definition of "template.")

  <Dimmable Button Data> "Huh?","Template»1"
 <Panel Prompt> "Make your choice, then click the right arrow."
<End Panel>
```

3. Then complete the "true" branch for saving a document in SimpleText with the panels you used when you revised to add dialog item coach marks (in Chapter 12). These panels should be the following:

```
# Revised for item coaches
<Define Panel> "3650"
<Format> "Body"
Using the SimpleText program on your hard disk, create the document you want to use as a template. Include all content that should appear in every copy.

When the template content is complete, save the document with a name such as "Template Original." (This is the version you can use to revise the template later.)

<Format> "Tag"
Do This
<End Panel>
```

<Define Panel> "5297"
* <Coach Mark> "CoachMark»1"*
<Format> "Body"
Now save a second copy of the document by choosing Save As from the File menu.

<Format> "Tag"
Do This
<End Panel>

 # Revised for item coach on stationery radio button in SimpleText
<Define Panel> "5582"
* <Coach Mark> "CoachMark»3"*
<Format> "Body"
In the dialog box, click the stationery radio button (on the lower right, with the icon that resembles a notepad) to activate it.

<Format> "Tag"
Do This
<End Panel>

#New for item coaches - number 1
<Define Panel> "7145"
* <Coach Mark> "CoachMark»4"*
<Format> "Body"
Type a name for the template.

<Format> "Tag"
Do This
<End Panel>

#New for item coaches - number 2
<Define Panel> "7236"
* <Coach Mark> "CoachMark»5"*
<Format> "Body"
Click Save.
The next time you open this document, the program will make a copy and you'll have to name it before you can work with it.

<Format> "Tag"
Do This
<End Panel>

4. Add the panels for the second branch of the task. First, create a new panel (with content similar to that in panel 3650, which was modified for the SimpleText branch) as follows:

 # New panel for second branch of radio button task
<Define Panel> "7912"
<Format> "Body"
Create the document you want to use as a template. Include all content that should appear in every copy.

When the template content is complete, save the document with a name such as "Template Original." (This is the version you can use to revise the template later.)

<Format> "Tag"
Do This
<End Panel>

5. Next, add a note to repeat panel 5297 after the new panel (7912):

 # Repeat panel 5297 here.

6. Finish the panels for this task with content from the original Info window version of the task:

<Define Panel> "7965"
<Format> "Body"
Check the Save dialog box to see if there's a Stationery option (usually in a Format pop-up menu). Choose Stationery if you find it.

Change the document's name and click Save.

<Format> "Tag"
Do This
<End Panel>

<Define Panel> "5745"
<Format> "Body"
If you did not use a Stationery format when you saved, locate the document's icon (in the Finder) and click to select the icon.

```
<Format> "Tag"
Do This
<End Panel>

<Define Panel> "6361"
  <Coach Mark> "CoachMark»2"
<Format> "Body"
Choose Get Info from the File menu.

<Format> "Tag"
Do This
<End Panel>

<Define Panel> "6470"
<Format> "Body"
In the Info window, click to place an X in the box labeled "Stationery
pad" at the lower-right corner of the window.

<Format> "Tag"
Do This
<End Panel>

# New panel for branching task
<Define Panel> "7518"
<Format> "Full"
The next time you open this document, the program will make a copy
and you'll have to name it before you can work with it.
<End Panel>
```

7. Go to the sequences section (VI) of the text file and replace the original "Creating a template" sequence with the branching version that you've seen already as follows:

```
# Revised for item coaches
# Revised again for radio button task
<Define Sequence> "Creating a template»1","Creating a template"
   <Seq Nav Button Set> "Start and Huh"
  <Panel> "2313"     #radio buttons added
  <If> radioButtonState("save a document as stationery using Simple-
Text","2313")
     <Panel> "3650"
     <Panel> "5297"
     <Panel> "5582"
     <Panel> "7145"
     <Panel> "7236"
```

```
  <Else>
    <Panel> "7912"
    <Panel> "5297"
    <Panel> "7965"
    <Panel> "5745"
    <Panel> "6361"
    <Panel> "6470"
    <Panel> "7518"
  <End If>
<End Sequence>
```

8. Save the text file, preferably with a new name that reflects its content.

9. Launch your Guide Starter application and compile the text file by choosing Compile Database from File from the File menu.

When your Guide is compiled, the radio button panel should look like the one in Figure 14-2.

Now that you're familiar with radio buttons and branches, you can move on to even more advanced features, such as AppleScript (Chapter 16) and context checks (Chapter 17). But first, sample another kind of intensive information management as you learn to prepare word lists for the Look For component of your Guide.

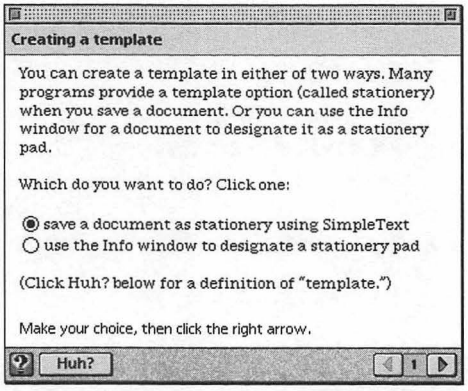


Figure 14-2
Radio button panel for a branching task

FIFTEEN

CREATING WORD LISTS FOR "LOOK FOR" ACCESS

Apple Guide provides three ways for users to find information in a Guide that uses the Full Access window. One of these methods, "Look For," allows the user to type a word or phrase that the Apple Guide extension processes and compares to the Index and other lists of terms. Figure 15-1 shows the Look For section of the Access window.

You can increase the effectiveness of the "Look For" feature in your Guide by adding the following lists of terms:

- ignore list
- exception list
- synonym list

These lists sharpen Apple Guide's searching capabilities. Without them, Apple Guide searches the Index for matches to the word or phrase a user types in the Look For text box. Apple Guide uses these added word lists to refine its search process. The rest of this chapter explains how they are used, and provides instructions for adding them to the text file for your database.

Ignore List

The ignore list provides a service like its name: It tells Apple Guide which words to disregard in a phrase typed by a user.

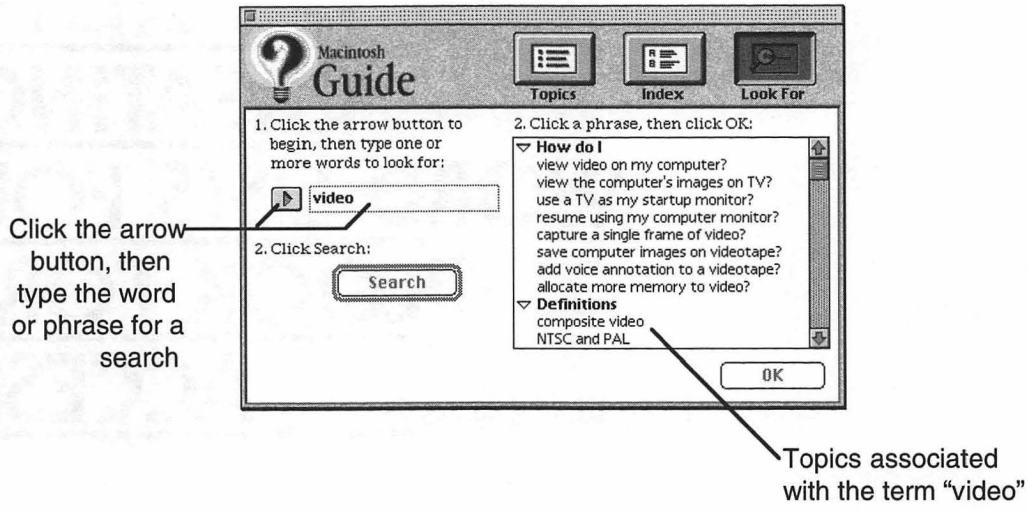


Figure 15-1
The Look For facility in which users enter terms they want to find

Basically, you want Apple Guide to pass over terms that do not relate to the Guide's content. Categories of words that should be included in an ignore list are contractions, adverbs, adjectives, and pronouns. Also included are articles, conjunctions, common verbs such as "is" and "are," personal terms such as "I" and "me," and most vague or collective terms, such as "those," "they," "who," and "why."

Figure 15-2 shows part of the ignore list for Macintosh Guide.

Exception List

To speed its searching, Apple Guide shortens most words to their roots—a process called "stemming." For example, stemming deletes word suffixes, including "al," "ed," "ies," "ing," "ion," "ize," and "s." Stemming also removes the last letter of words that end in double letters.

In a few instances Apple Guide's search engine adds an "e" to the end of a word after it removes the suffix; for example, it changes "riding" to "ride." But a similar word, "pasting," is stemmed to "past," and no "e" is added at the end, thereby altering the word's meaning.

Sometimes stemming results in a word whose spelling matches something in the Index but whose meaning is entirely different. To

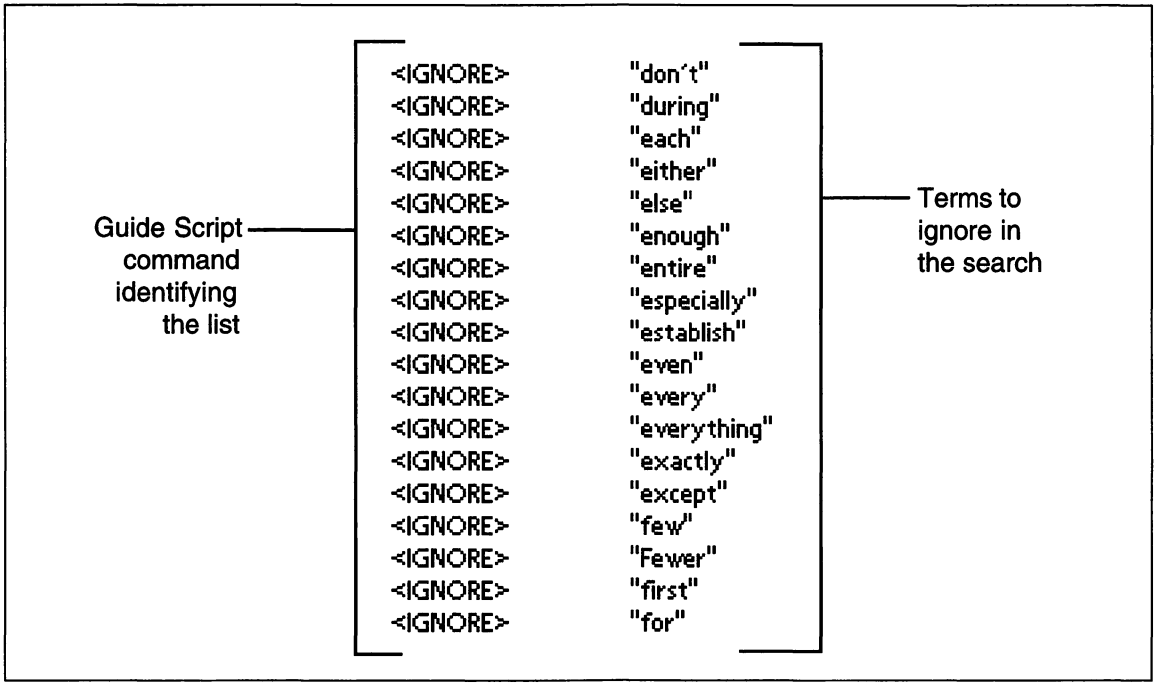


Figure 15-2
Excerpt from an extensive ignore list

avoid such problems, you can provide an exception list, which tells Apple Guide not to stem the words it contains.

Figure 15-3 shows an excerpt from the exception list for Macintosh Guide.

These are terms either that are likely to be made incorrect when stemmed by Apple Guide or whose stemmed version has a different association with a Topic than the longer version does. To ensure that both words keep their associations with Topics, Apple Guide must skip over these words when it performs a search.

Synonym List

A synonym list can significantly enhance the success of a user's search. You include in it terms that are synonymous with words in the Index. This is an area in which you can get help from others, for example, by asking what words they would look for in connection with particular tasks.

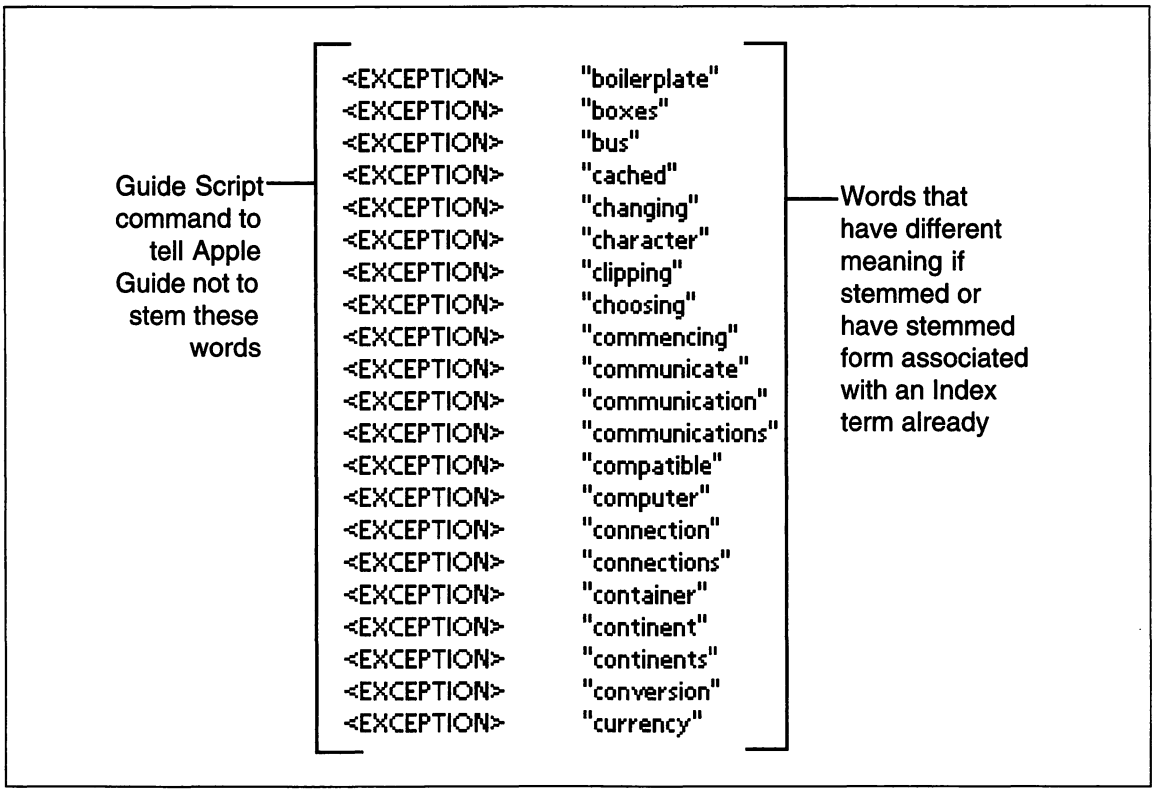


Figure 15-3
Exception words are not stemmed by Apple Guide in a search

Keep in mind users of other types of computers when you create a synonym list. For example, Macintosh computers are frequently used on networks with DOS computers, and users of both types of machines frequently exchange files and information. So try to get lists of suggested terms for your Index or synonym lists from DOS users as well as Macintosh users.

Figure 15-4 shows sets of synonyms for two Index terms in Macintosh Guide.

Enter Word Lists

You can keep track of synonyms and begin working on exception and ignore lists as you enter Index terms. The ideal time to brainstorm the contents of the synonym list is while you're writing the content of Topics, because then you are likely to be immersed in the subject and its terminology.

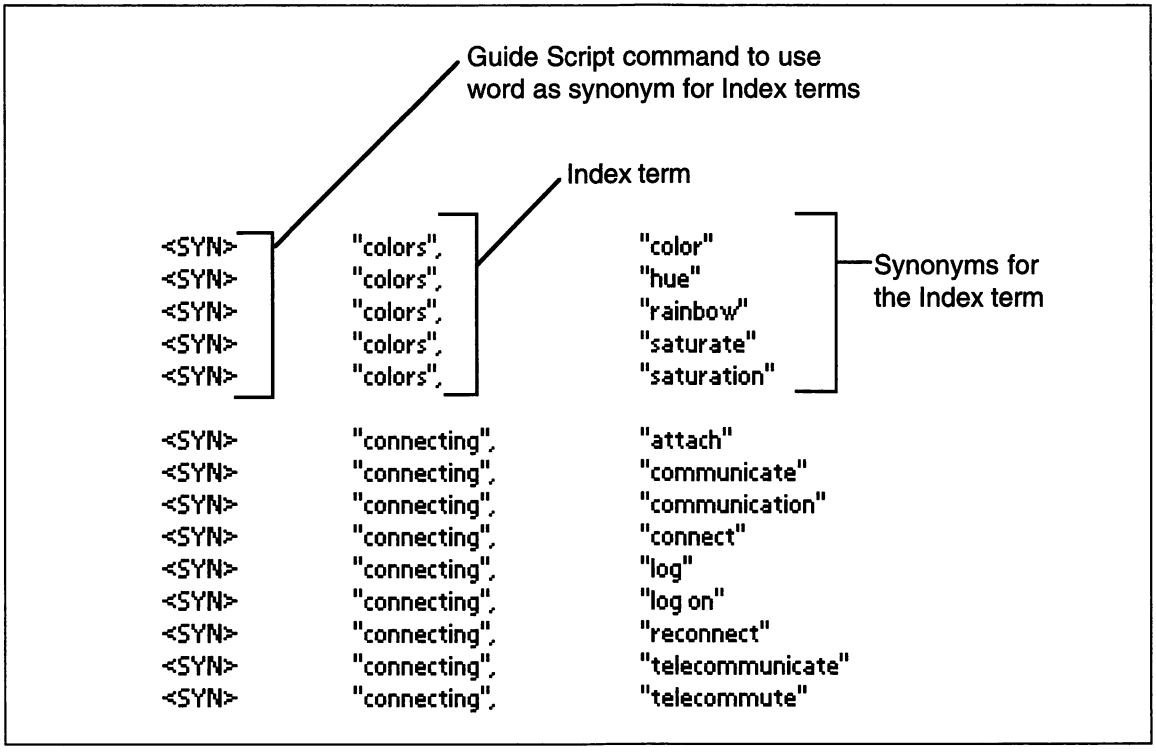


Figure 15-4
Part of the synonym list for Macintosh Guide

You can easily enter the three word lists in your Guide's text file. Section IX is designated for that content. Here's the format for each type of list:

```
<IGNORE>    "our"
<EXCEPTION>    "option"
<SYN>    "mouse"    "pointer"
```

In the synonym example, "mouse" is the Index term and "pointer" is the synonym.

The size and complexity of your Look For lists will depend on the amount and variety of material in your Guide. Regardless of your Guide's size, however, it and its users will benefit from having these three word lists.

SIXTEEN
USING APPLESCRIPT

When you browse Apple Guide databases that come with a newly purchased Macintosh or System 7.5 upgrade, you are probably intrigued by the apparent communication between the Guide and applications such as the Finder. For example, when you follow through the steps to learn how to adjust the sound volume of your Macintosh, how does Apple Guide open the Control Panels folder for you if you don't perform the step yourself? Once the window is open, how does Apple Guide know where on your screen to draw the coach mark for the Sound control panel icon?

The "magic" that goes on behind the scenes is Apple's system-level scripting environment (called OSA, for Open Scripting Architecture). The most popular language for system scripting is AppleScript. Combining Apple Guide and AppleScript gives your Guides the "active assistance" your users will appreciate.

About AppleScript

AppleScript scripts are usually short English-like programs that automate very specific tasks with the Finder or other scriptable applications. For example, System 7.5 comes with a number of

prewritten scripts (installed into the Automated Tasks folder in the Apple menu). One script lets you drop any icon on it to place an alias to that item into the Apple Menu Items folder. That script is merely a series of instructions that tells the Finder to make an alias of the item and move it to the Apple Menu Items folder.

Not all parts of the Finder are scriptable (the contents of most control panels are not, for example), nor are all applications, although the list of scriptable programs is increasing rapidly. Popular programs, such as Microsoft Excel 5, Microsoft Word 6, WordPerfect 3.1, Quark XPress 3.3, and FileMaker Pro are scriptable.

AppleScript's Components

Standard system software installation places a number of AppleScript-related items on your hard disk. The most important is the AppleScript extension (Extensions folder). Another extension, called Finder Scripting Extension, is the file that makes the Finder in your Macintosh scriptable. Also inside the Extensions folder is the Scripting Additions folder. Items there add to the vocabulary of the basic AppleScript language. The other vital element is a program called Script Editor.

Script Editor—Where You Write Scripts

System 7.5 comes with everything you need to write scripts, provided you have some help from outside sources to learn the details of the language (see "AppleScript Information Sources" at the end of this chapter). Virtually all script-writing activity takes place in the Script Editor (or third-party script editor). That's where you can experiment with examples in this chapter.

Once you write a script that works as you expect, you can save it as a compiled script or a stand-alone double-clickable application. A compiled script can be run only from the Script Editor program. For Apple Guide purposes, a compiled script is all you need. The Guide compiler grabs a snapshot of the script and embeds it into the Guide database.

How Apple Guide Uses Scripts

AppleScript scripts help a Guide perform two valuable tasks:

- locate an item on the screen for coach marks when the item may be in a different spot each time
- automate a task for users.

Despite the scripting that these two tasks have in common, they are very different parts of Apple Guide.

Scripted Coach Marks

In Chapters 6 and 12, you saw how Apple Guide allows various categories of coach marks to be used in windows, menus, and on items in dialog boxes. Window, menu, and item coaches are defined by fixed items in a program. Locations of all these items—a button in a window, a menu item, or a field in a dialog box—remain the same each time the program runs on all Macintosh screens. Because window-related items, such as fields and buttons, are defined relative to the geography of the window, then as long as the correct window is showing, the coach marks will always be where the items are. Therefore it is a simple task to define the coach mark around such known items.

In some cases, however, an item for which you want a coach mark may not always be in the same place. Macintosh users have monitors of all shapes and sizes. Some monitors display 640 pixels across; while others show as many as 1280. So it is impossible to designate a fixed location for a coach mark that needs to circle the startup disk icon near the top right-hand corner of the screen. But an AppleScript script can ask the Finder to supply the necessary coordinates of that icon (top left and bottom right corners) the instant the information is needed, regardless of the monitor's size or the location of the icon.

What Apple Guide Needs for an AppleScript Coach Mark

Scripted coach marks are treated no differently in your Guide text file than other types of coach marks. You must define the coach mark and call that coach by name in the panel definition. The syntax for an AppleScript coach mark definition is

<Define AppleScript Coach> coachMarkName [, coachStyle] , AppleScriptID

The *coachMarkName* parameter is the unique name for this particular coach mark, the name your panels will use to invoke this coach mark. The *coachStyle* parameter is optional (REDCIRCLE is used when this parameter is missing), and all possibilities available for other coach mark styles apply here (see Chapter 12 for other styles).

The final parameter, *AppleScriptID*, is the heart of this definition. Its value can be a resource ID for the script, provided you've turned a script into a resource that you load with a corresponding <Resource> command in your text file. More commonly, however, it is the name of the compiled AppleScript file. You can use just the filename if the script is in the same folder as Guide Maker or a path name relative to Guide Maker if you keep your script stored separately (recommended later in this chapter).

When a panel appears that calls this coach mark, Apple Guide runs the script to obtain the coordinates for the coach mark. Therefore the script must be written so that it returns a valid set of coordinates. If the returned value comes back as empty, then Apple Guide does not draw the coach mark.

Writing a Script That Returns Coordinates

The following discussion assumes you have some exposure to AppleScript. Even if you've just dabbled in other programming, such as HyperTalk, you should be able to understand the concepts.

There is no distinction in AppleScript between a command script, which carries out some action and goes away (some languages call it a *procedure*), and a function, which may perform an action but which always supplies information back to whatever called it. In AppleScript, a script is a script. Whether a script returns a value is determined by the presence of a Return statement. AppleScript-style coach marks expect a returned value in the form of coordinates—a series of four integers representing the left, top, right, and bottom pixel counts of a rectangular area on the screen. AppleScript groups these values in the form of a list that is a comma-delimited series of items inside curly braces. Here is an AppleScript list for the rectangle of a 640 x 480 pixel screen:

```
{0,0,639,479}
```

Like many programming numbering schemes, Macintosh screen coordinate systems start with zero. However, the joy of AppleScript's providing coordinates for your coach marks is that you never have to worry about the specific numbers.

The Finder and many scriptable applications maintain properties for objects you see on the screen. One of the most common is one called bounds, meaning the bounding rectangle of the object. The value for this property is already in the desired list format. The simplest script that supplies this information about the startup disk is as follows:

```
tell application "Finder"
    return bounds of startup disk
end tell
```

The startup disk (icon) is an object defined in the Finder, and it has the bounds property.

Advanced Concerns

While the previous simple script supplies the information required for the coach mark, you will want to give your script some more intelligence, depending on where you anticipate your Guide's users to be when the panel displaying this coach mark appears. This is part of the planning that must go into high-quality Guides.

The following script demonstrates what we mean. This script does a more complete job of preparing the icon for the coach mark (the ¬ symbols are line continuation symbols to let long lines fit within the margins of our page):

```
tell application "Finder"
    if (count windows) ≠ 0 and ¬
        name of window 1 is startup disk then return
    if application "Finder" is not frontmost then activate
    if (count windows) = 0 or name of window 1 ¬
        is not the name of the startup disk ¬
        then return bounds of startup disk
end tell
```

This script (and its panel) is intended to provide a coach mark on the startup disk when the task requires the startup disk to open. The

second line of this script does some status checking before going to the trouble of preparing the icon for its coach mark. This coach mark need not be shown if there is at least one window open on the Desktop and its name is that of the startup disk. In other words, if the startup disk is open, then the script ends (it returns nothing), in which case the coach mark doesn't appear.

If processing continues, then the Finder's front window (if any) is not the startup disk's window. The next step then is to make the Finder the front-most application, which users will need anyway in order to carry out the step detailed in the panel. Another status check follows, this time to return the rectangle value of the icon if either of two conditions applies: either there are no windows open or the front-most window of the Finder is not the startup disk. In other words, the script returns the rectangle value only if the icon needs double-clicking (either to open the window or to bring it to the front).

Working with coordinates can be a bit tricky when the Finder normally returns values based on a particular window because coach mark coordinates must be based on the screen geography. To convert coordinates of an item within a window to screen coordinates, your script must take into account the bounds property of the containing window. The following script returns the screen coordinates of the Color control panel icon if the Control Panels window is not open:

```
tell application "Finder"
    if (count windows) ≠ 0 and name of window 1 is "Color" then return
    reveal control panel "Color" of control panels folder
    activate
    copy bounds of selection to iconRect
    copy bounds of window "Control Panels" to windowRect
    set iconsRect to ((item 1 of windowRect) + (item 1 of iconRect)) ¬
        & ((item 2 of windowRect) + (item 2 of iconRect)) ¬
        & ((item 1 of windowRect) + (item 3 of iconRect)) ¬
        & ((item 2 of windowRect) + (item 4 of iconRect))
    if name of window 1 is not "Color" then return iconsRect end tell
```

After bringing the Finder to the front, the script captures the bounds of both the icon and the Control Panels window. The long Set state-

ment then adds the window bounds values to the corresponding values of iconRect to come up with screen coordinates. For example, if the left coordinate of the window is 100 (with respect to the screen) and the left coordinate of the icon is 10 (with respect to the window), then the icon's left coordinate with respect to the screen is 110.

Don't be put off by what may appear to be complex logic in the sample scripts. Many of the status check statements in these kinds of scripts come as a result of user testing of Apple Guide; a coach mark may appear when it doesn't need to, or it doesn't appear when it should because the user is in a state other than the ideal one. Also, AppleScript coach marks tend to perform similar actions on different Finder elements. Experience you gain from writing one script will pay off handsomely in the next one you write.

Automation Scripts

The second blending of Apple Guide and AppleScript comes when you design a Guide to offer assistance to users who don't follow the steps detailed in the Guide or when you offer a "Do it for me" button to let the Guide carry out a complex process. This kind of assistance usually comes in Tutorial Guides and in any Guide sequence that answers a "How do I?" question. For example, if your Guide were to contain a sequence that directs users to open the System folder, one of the first tasks would be to open the startup disk. The panel that contains that instruction might call the AppleScript coach mark described in the previous section to circle the startup disk icon. If a user continued to the next panel without actually opening the startup disk icon, the Guide would perform a context check (Chapter 17) to see if that window were open; if it wasn't, an intervening panel would tell users that Apple Guide is opening the startup disk automatically. We reserve until Chapter 17 the details about how to integrate context checking and AppleScript scripts, so we confine our discussion here to the scripts you might call from a Guide.

Perhaps the most common kind of script is one that opens something that the Guide users failed to open in a Guide step. In fact, virtually every script used in the Macintosh Guide that ships with System 7.5 has a name that begins with "Open." This is because the scripts automatically open something like a control panel or folder.

An example of a script whose job is to open the Apple Menu Items folder follows:

```
tell application "Finder"
    if application "Finder" is not frontmost then activate
    open apple menu items folder
end tell
```

The scriptable Finder predefines the objects and properties that make scripts like this one simple to understand and write. This script checks whether the Finder is the front-most application; if it isn't, the script brings it to the front so that users will see the remaining activity. A simple Open command to the desired folder opens the folder directly without going through the longer series of steps (open the startup disk; open the System Folder). The scriptable Finder allows script control over many of these kinds of operations.

It would be a mistake, however, to limit the powers of such assistance to merely opening Finder objects. Anything you can script in a scriptable application is fair game, provided it contributes to teaching a task to the Guide's users (automation scripts perform actions but do not return any values). Just how far a script may go to assist users is largely defined by the level of scriptability of the application(s) covered by your Guide. HyperCard stacks or stand-alone applications, for example, are highly scriptable such that a script can perform button clicking, data entry into text fields, and menu choices. In an entirely different scenario, your Guide may open a detailed help document outside the Guide that contains in-depth reference material (perhaps in DocViewer or Acrobat format). The associated script could print that file for users.

The depth of the program's scripting dictionary (openable from the Script Editor's File menu) and the quality of the scripting implementation by the program's author govern the powers of your automation scripts. Therefore your Guide may be inconsistent at times because it is capable of assisting in places that have good scripting support but not in those that don't. However, remember that a Guide's job is to teach (or at least rekindle a memory) and not to become a separate user interface for a program or task. The best way to learn a task is by doing it, not by watching it being done all the time.

Managing Script Files

A script you write for automated tasks is saved in the same way as an AppleScript coach mark script. The script's name becomes a parameter of the <On Panel Show> Guide Script command, as demonstrated in Chapter 17.

If you begin to write a lot of scripts for your Guides, the folder containing Guide Maker will get pretty crowded. So it is best to create some subfolders for the script files. One strategy is to create one general AppleScript folder and then add a subfolder dedicated to scripts for each Guide you work on.

Whenever a command in the Guide text file calls for the *AppleScriptID* parameter (in the <Define AppleScript Coachmark> and the <On Panel Open> commands), you need to supply a path name to the script relative to the location of the compiler program. Begin the notation with a colon, followed by the colon-delimited path name to the script file from the compiler's folder. (The initial colon essentially takes the place of the name of the folder in which the compiler is located.)

For example, assume you have a script named "Open Preferences" stored in a folder named "E-Mail Guide AppleScripts," and that folder is in the AppleScripts folder. The compiler program is in the same folder as the AppleScripts folder. The *AppleScriptID* parameter would be filled by the phrase in Figure 16-1.

Important: The path name that directs Guide Starter's compiler program to an *AppleScriptID* parameter must always begin with the location of the compiler (preceded by a colon). The program does not accept path names that begin with any other location (even the root level of the hard disk).

Adding Scripts to Guide Starter Text Files

While you don't write the AppleScript scripts in the text file, some of the commands will call the saved scripts by name. In the Guide Starter text file, you would add an AppleScript coach mark definition to the coach marks section (IV). Automation scripts are called from panel definitions, so you would write those panels in Section V.

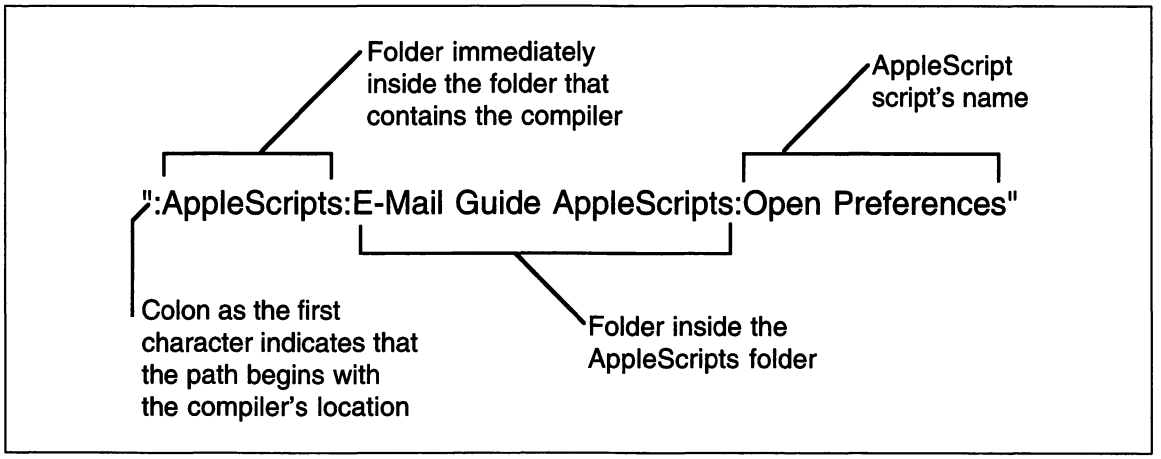


Figure 16-1
Path name format the compiler uses to locate an AppleScript script

Near the bottom of the text file, Guide Starter creates Section X for AppleScripts. It does this so you can write commented lines about the names and purposes of scripts you define. Then, if you want to reuse an existing script for another panel, you can refer to your notes in Section X, rather than having to sift through the various script files.

AppleScript Information Sources

In-depth coverage of AppleScript is beyond the scope of this book, but several books are available that do provide such coverage, including the following:

- *Danny Goodman's AppleScript Handbook*, Second Edition (Random House, 1995) comes with a diskette full of scripting examples for learning basic scripting and working with the Finder and popular scriptable applications.
- *AppleScript Language Guide* (Addison-Wesley, 1994) is the primary language reference from Apple Computer, Inc.
- *AppleScript Scripting Additions Guide* (Addison-Wesley, 1994) is Apple's reference manual for its own scripting additions.

- *AppleScript Finder Guide* (Addison-Wesley, 1994) is Apple's official technical reference for scripting the Finder.
- "Using AppleScript" (parts 1 and 2) is inside the Apple Extras folder installed as part of System 7.5. This folder also contains the AppleScript Script Editor and two sets of scripts that automate some common tasks, such as turning the computer's speaker on or off. These scripts are located in the folders Automated Tasks and More Automated Tasks, both inside the Apple Extras folder.

SEVENTEEN

USING CONTEXT CHECKS

When extended to its fullest powers, an Apple Guide seems to be watching over the user's shoulder. For example, new users may follow the steps of a tutorial Guide to learn a new task. Along the way, they may not actually carry out a step, or may not know how to do it, or may accidentally click someplace that hides the active window. A tutorial that proceeded to the next step despite these errors would leave its befuddled users in the dust.

To prevent these learners' nightmares, Apple Guide features context checking, the ability to examine the state of windows, menus, dialog boxes, and various system settings. A tutorial Guide outfitted with context checking ensures that users have completed one step before they proceed to the next.

When to Use Context Checks

There is no rule that says a Guide must have context checking. Much depends on the purpose and design of your Guide. Context checks can be most helpful to your Guide's users when the Guide leads them through steps to accomplish a task. Tutorials are obvious candidates for context checks within the sequences that provide step-by-step instructions.

Other good candidiates are more general Guides that include panel sequences that lead users through steps. Typical situations would be sequences linked to "How do I?" kinds of questions in the Access window. If you find yourself building lots of panels in the Tag format with the Tag reading something like "Do This," then you should investigate building context checks into those sequences.

Context Check Elements

When a sequence performs a context check, it essentially asks a question that can be answered true or false, yes or no. For example, it might ask if a particular window is open or a menu item is checked.

Like most features of an Apple Guide, context checks must be defined in the Guide text file. Once they are defined, your sequences may summon them by name. Calls to context checks are made as parameters to a few Guide Script commands (described later in this chapter). The outcome of the context check then determines the flow of the panels within the sequence the user sees.

There is one additional piece to the context checking puzzle: resources that perform the actual work of communicating with a program to determine whether the program is in the state sought by the context check. For the most common types of context checks, the loading of these resources is automatic when the Standard Resources file (supplied with Guide Starter) is loaded.

Types of Context Checks

To assist Guide authors in writing context checks, Apple engineers have provided a number of standard resources (of type 'extm', for external module) that perform the actual communication with the program. Additional external modules are available on the compact disc (CD) that accompanies *Apple Guide Complete*. If you are creating a Guide for an application developer and need further methods of context checking, you can consult the programmer about creating the necessary 'extm' resource (examples are on the CD, too).

Before getting into the syntax for defining context checks, following are some of the most common types of checking your Guides can do with the standard resources supplied by Apple (they are grouped by the external module that contains them):

Windows

- A particular window in a particular program is open.
- A particular window in a particular program is active.
- An Info-style window is open.
- The active window is an Info-style window.
- There is an active window on the desktop.
- The active window is set for file sharing.
- The Find window is active.
- The extended Find window is active.

Dialogs

- A particular dialog (by resource ID) is active.
- There is an active dialog box on-screen.

Menus

- A particular menu exists.
- A particular menu item exists in a particular menu.
- A particular menu item in a particular menu is checked.

Files

- A particular file is in the Extensions folder.
- A particular file is in the System Folder.
- A particular file is in the Apple Menu Items folder.
- A particular file is in the Control Panels folder.
- AppleShare is installed on this Macintosh.
- PrintMonitor is installed.
- QuickDraw GX is installed.
- This Macintosh has a RAM disk running.
- Startup disk window is the active window.
- Startup disk window is open.
- Control Panels window is active.
- System Folder window is active.

- Apple Menu Items folder window is active.
- Apple Menu Items folder window is open.
- Extensions folder window is active.
- Extensions folder window is open.

System Readings

- File sharing is on.
- File sharing is off.
- File sharing guest access is off.
- Video monitor bit depth is greater than some value.
- Video monitor bit depth is at least some value.
- Video monitor is monochrome only.
- Multiple video monitors are connected to this Macintosh.
- CPU is in the 68000 family.
- CPU is a 68000.
- CPU is a 68030 or greater.
- CPU is a 68040.

Chooser Readings

- A particular printer is the currently selected printer.
- A non-networked printer is selected.
- A networked printer is selected.
- No printer is selected.
- AppleTalk is off.
- The network has multiple zones.

Processes (Items in the Application Menu)

- A particular application is active.
- A particular application is open.

Most of these external modules were crafted in response to the needs of the authors of Apple's own Guides for System 7.5. While many are useful to Guides covering other topics, there may not be a

context check for every situation you encounter while creating your Guides. For example, if your Guide would like to check the setting of a push on–push off button in an application window's tool bar, there is no standard context check for that item. Perhaps when that button is engaged, a menu item is checked—something you can context-check. We'd like to see application developers produce and distribute free of charge external modules that allow third-party Guides to perform specific context checks within their programs.

Defining Context Checks

Before you can call a context check from a sequence, the context check must be defined in the Guide text file. If it weren't for existing context definitions supplied by the authors of the external modules, it would be quite difficult for a Guide author to write a context check definition. The precise parameters for the <Define Context Check> Guide Script command depend entirely on the construction of the 'extm' resource—the purview of programmers.

For the scope of this book, we provide context check definitions for a handful of the most common context checks you're likely to try out with Guides you create with Guide Starter. To use any of these context checks, simply enter the commands into your Guide's text file in the context check section (XII). For the sake of brevity, we use the Guide Script abbreviation <DCC> for the <Define Context Check> command.

Figure 17-1 shows a selection of the context checks built into Apple Guide (and provided in the Standard Resources file on the Starter Kit disk).

Despite the apparent complexity of these definitions, most of the parameters are already set in the definitions, thereby usually leaving just one or two to be supplied when they are called from a sequence. While we haven't yet shown you where in the Guide's text file the calls are made to the context checks, the examples in Figure 17-2 include a call to each of the context checks shown in Figure 17-1.

Each of these context check calls returns either a true or a false value, which is required by the decision-making Guide Script commands that incorporate these calls (below). Most of the parameters in the figure are simply the names of menus, menu items, and windows.

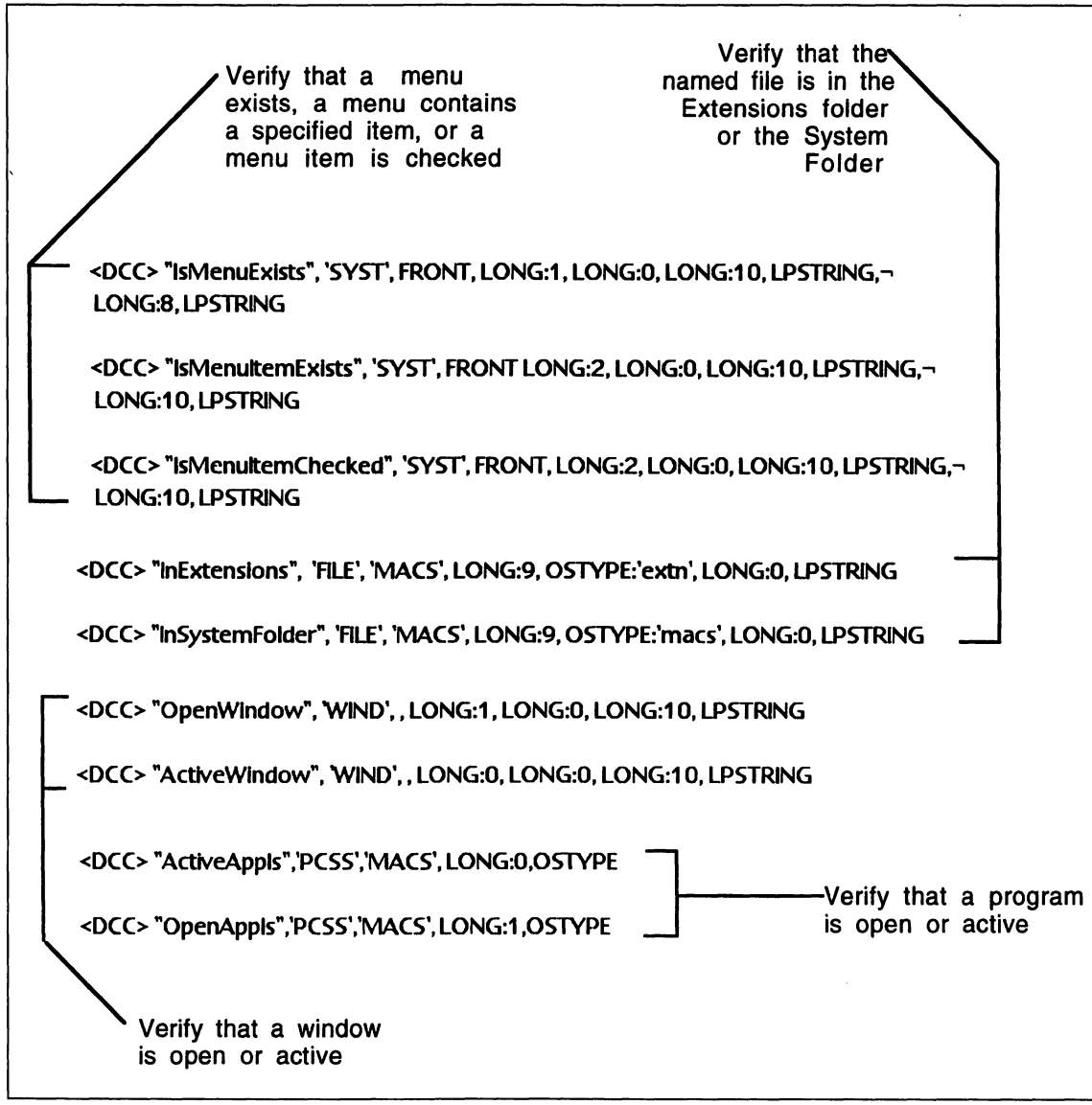


Figure 17-1
Definitions for some of Apple Guide's built-in context checks

If you're not familiar with the signature of an application, it is the four-character identifier that the Finder uses to keep applications and their documents together. Application signatures of all commercial and popular shareware products are registered with Apple Computer to prevent duplication. You can use the AppleScript Script Editor

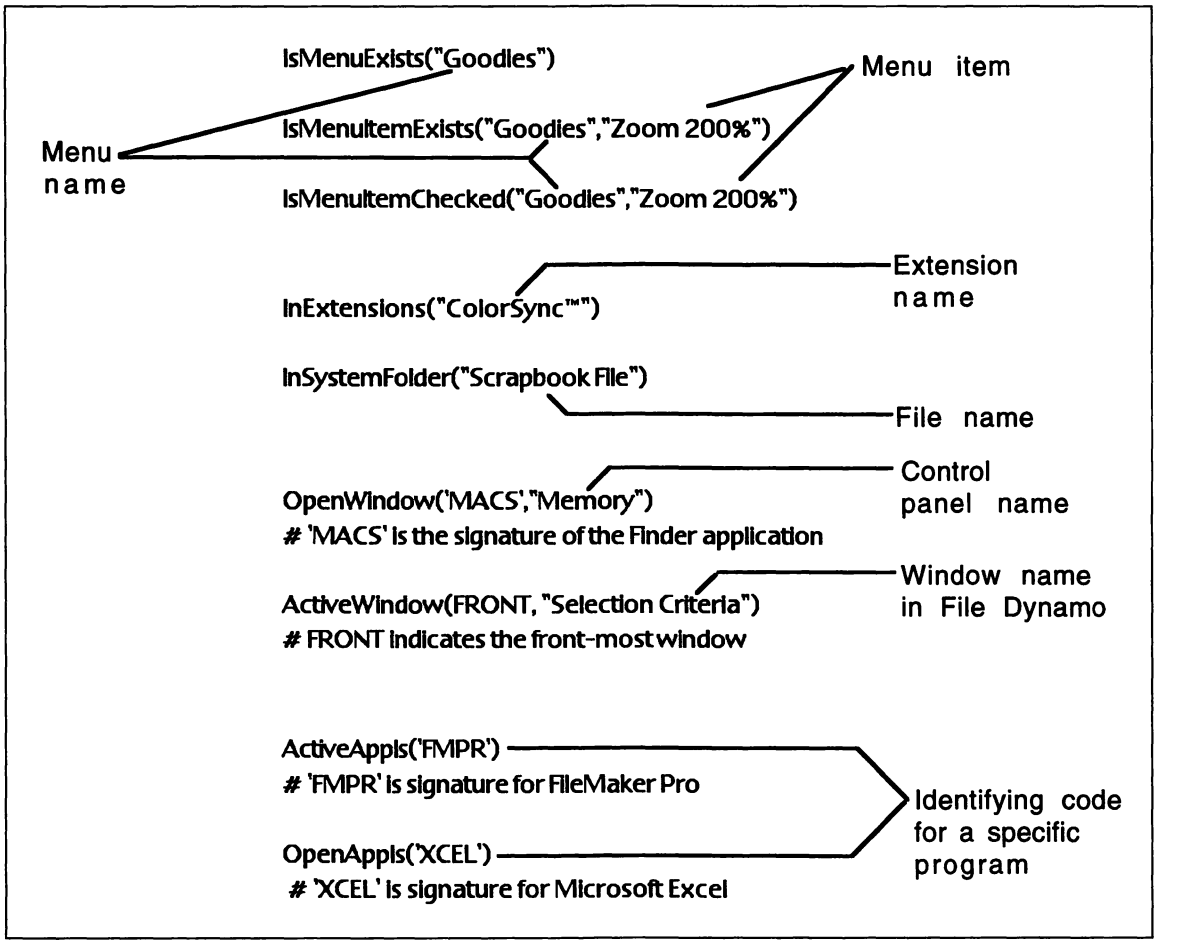


Figure 17-2
Examples of Guide Script calls to context checks

(provided with System 7.5 in the Apple Extras folder) to learn the signature of an application. Enter and run the following one-line script:

file creator of (info for (choose file))

In the file dialog that appears, choose the target application. The Result window will display the signature value needed for some of the previous context checks. Be cautious of two factors:

- Signatures are case-sensitive, so copy a signature exactly as it is returned by the script.

- Signature parameters must be in straight single quotation marks; while other names of items are in straight double quotation marks.

We demonstrate a couple of these context checks later in the chapter and give you some ideas to test out with File Dynamo.

Guide Paths

Context checking is designed to direct the user's journey through a Guide sequence and especially to prevent the user from ending up in a blind alley. As we've said elsewhere, a successful guide is one that anticipates every possible misstep the user could make along the way. Context checking (with or without AppleScript scripting components) helps to keep your user on track.

Unlike decisions made by the user in a radio button panel (see Chapter 14), sequences containing context checks make seemingly automatic panel navigational decisions for the user based on the state of windows, menus, applications, and so on. While Apple Guide provides much flexibility in the way you can set up navigation paths for the user, we next highlight three of the most common ones.

Make Sure➠Oops

Scenario: A user wants to search for text in the current document but doesn't know how to start the process. Consulting the Guide, he selects an item that says "How do I search for text?" He reads a panel containing instructions to choose Find from the Edit menu. He doesn't follow the instruction and clicks the Guide's right arrow. The Guide sequence is written to make sure that the Find window is visible before showing the next panel. When the context check for that open window fails, the Guide instead displays an Oops panel (with a lone OK button) alerting the user that the Find window isn't open. If the user clicks the OK button without opening the Find window, he goes back to the panel that tells him again about choosing the item from the Edit menu. But if the user opens Find and then clicks OK, the Guide sequence continues to the next step in performing a search.

This is one version of a Make Sure conditional statement that goes into a sequence definition. Syntax for this command is

<Make Sure> condition, oopsOrContinueSequenceName

The *condition* parameter in this scenario is the context check that sees if the Find window is active. Whenever the *condition* parameter returns false (for example, the Find window is not active), then execution branches to an entirely different sequence whose name is the second parameter of the command.

Because the design for this context check branches to a one-panel Oops sequence (the panel is conveniently defined inside the sequence definition), we call this a Make Sure➠Oops context check. Here is a Guide Script segment that demonstrates the previous scenario:

```
# Regular sequence that leads to the Find window
<Define Sequence> "How do I search for text?»1","How do I search for
text?"
 <Panel> "Find overview"
 <Panel> "Choose Find in Edit menu"
  <Make Sure> ActiveWindow("Find"),"Oops: Missing Find window"
 <Panel> "Enter text to search"
 <Panel> "Click Find button"
<End Sequence>

# Sequence for Missing Find window Oops panel
<Define Sequence>     "Oops: Missing Find window"
 <Sequence Prompt Set>          NONE
 <Define Panel>          "Oops Panel: Missing Find window"
  <Format> "Tag"
Oops

  <Format> "OopsBody"
The Find window is not activated. Click OK for instructions (or click
inside the window to make it active, then click OK).

  <Standard Button>    "OK", Center, GoBack()
 <End Panel>
<End Sequence>
```

This second sequence is a self-contained unit that handles the entire Oops situation. There is a panel defined in the OopsBody format (Guide Starter defines this for you in the text file), including a button

whose definition is built into Apple Guide. That button performs a GoBack event (predefined for you by Guide Starter), the crux of how this panel behaves. Remember that execution reaches this sequence from the <Make Sure> command. The GoBack event sends Guide execution back to that spot, where <Make Sure> checks the context once again. If everything is OK, the user can continue; otherwise, the user sees the panel from the previous line of the sequence.

Oops panel content has the important job of telling the user what went wrong and how to get back on track. More experienced users will know what went wrong and will try to recover while the Oops panel is showing. Newcomers will appreciate the Oops explanation and the helping hand back to the previous instruction.

Make Sure➠Automated Scripting

Scenario: A user reads a panel containing instructions to choose Find from the Edit menu. She doesn't follow the instruction and clicks the Guide's right arrow. The Guide sequence is written to ensure that the Find window is visible before showing the next panel. When the context check for that open window fails, the Guide displays a panel alerting the user that the Find window isn't open and that the Guide is opening it for her. When the window appears and the user clicks the Continue button, the Guide sequence continues to the next step in performing a search.

This scenario has much in common with the Make Sure➠Oops scenario. But in this case, the Guide doesn't let the user make the same mistake or omission twice. When it sees that a step hasn't been followed, it summons an AppleScript script to make the menu choice, while telling the user what went wrong and what it's doing to rectify the situation. We call this decision method a Make Sure➠Automated Scripting context check.

This context check and the Make Sure➠Oops context check differ mainly in the branch sequence that executes when the <Make Sure> check fails. As before, the second parameter of the <Make Sure> command is the name of the sequence to which to branch. But this type of sequence has different features. Here's an example that could fit our scenario:

```
# Sequence to automatically choose the Find menu
<Define Sequence> "Auto menu: Do Menu Item Find seq"
 <Seq Nav Button Set> NONE
 <Define Panel> "Auto menu: Do Menu Item Find"
  <Format> "Full"
Please wait a moment. Apple Guide is assisting you by choosing Find
from the Edit menu.

   <3D button> 1070,1072,Center,GoBack()
   <On Panel Show>  DoAppleScript(":AppleScripts:Do Menu Find")
   <Panel Prompt> "Item Opened"
 <End Panel>
<End Sequence>
```

For this sequence to compile properly, you must take care of two items. First, this panel executes an AppleScript script, named "Do Menu Find," which must be written with Script Editor. For our example, we assume that the program for which we're writing the Guide allows making a menu choice via an AppleScript script (many do). As indicated by the <On Panel Show> command, the script is executed immediately after the panel defined here appears on-screen. Notice that the path name we've supplied for the script indicates that it is in an AppleScripts folder located at the same level as the Guide compiler.

The second item that needs attention is the panel prompt. This example implies that a new prompt set has been defined in the Guide's text file (in Section III). Prompt text for this kind of panel should advise users to wait until the item is opened before clicking the Continue button.

As for the 3D button, the first two parameters of the <3D button> command are resource IDs for the up and down versions of the Continue button. These resources are automatically loaded into your Guide from the Standard Resources file.

AppleScript scripts called as the result of a failed context check can perform double duty to help the user out of inadvertent jams. If a user accidentally deactivates the target program while using the Guide, the results of the script execution may not be visible, and the Guide will run into problems further down the sequence. But all scripts can check for whether the target application is the front-most application and activate it if it is not. Thus the script both activates the program and performs the scripted action to satisfy the context check. This is much cleaner than performing an additional context check (for active application) in the Guide.

Your decision to employ a Make Sure➠Automated Scripting context check will often be influenced by how scriptable an application is (see Chapter 16). If you are working with another person who writes the scripts, determine early on what your ideal scripting would consist of. Then consult with the scripter to see if your wishes can be fulfilled. Don't be surprised if you must rethink some decision paths when some tasks cannot be scripted the way you'd like. You may have to fall back to a Make Sure➠Oops construction instead.

Skipping Unneeded Panels

Scenario: A user is viewing the Find window and wants to know what to do next. Consulting the Guide, he selects an item that says "How do I search for text?" The Guide detects that the user has already reached the Find window, so there's no need to display the introductory panel or the one that supplies the menu instruction. Instead, the Guide immediately goes to the panel that talks about the Find window.

What's special about the sequence executed here is that it knew when to skip over panels that the user didn't need. A Guide Script command, <Skip If>, handles this for you. Here's the syntax of this command:

```
<Skip If> condition
```

The *condition* parameter is a context check, just like the ones described for the <Make Sure> command. When execution of a Guide sequence encounters this command, Apple Guide checks the validity of the context check; if the returned value is true, then execution passes over the next line and continues immediately with the subsequent line.

The sequence for our scenario would be as follows:

```
<Define Sequence> "How do I search for text?»1","How do I search for
text?"
  <Skip If> ActiveWindow("Find")
   <Panel> "Find overview"
  <Skip If> ActiveWindow("Find")
   <Panel> "Choose Find in Edit menu"
  <Panel> "Enter text to search"
  <Panel> "Click Find button"
<End Sequence>
```

Because the <Skip If> command affects only one panel, we issue the same command with the same context check twice in a row. A sequence that contains more steps might have a variety of context checks in different places, each one testing whether the next panel's step has already been carried out.

In practice, you will typically combine <Skip If> and <Make Sure> commands within a sequence. While the <Skip If> assumes an advanced user, the <Make Sure> strings a safety net for the novice. The main sequence for our scenario that combines the Skip and Make Sure➡Oops context checks would look like this:

```
<Define Sequence> "How do I search for text?»1","How do I search for
text?"
  <Skip If> ActiveWindow("Find")
   <Panel> "Find overview"
  <Skip If> ActiveWindow("Find")
   <Panel>"Choose Find in Edit menu"
  <Make Sure> ActiveWindow("Find"),"Oops: Missing Find window"
  <Panel> "Enter text to search"
  <Panel> "Click Find button"
<End Sequence>
```

We make three different calls to the same context check, all of which are necessary to provide an intelligent Guide to the user. When the sequence involves even more steps, each of which depends on the previous one being completed, the interweaving of <Skip If> and <Make Sure> commands can get a bit complex unless you carefully follow the logic along the path. For hundreds of examples, consult the text files for the Macintosh Guide supplied on the CD-ROM that accompanies *Apple Guide Complete*.

Incorporating Context Checks into a Guide's Text File

What may seem a bit overwhelming at first is that there are many components to context checking, components that go in different places within a Guide's text file. The division of the Guide Starter text file into section numbers helps you sort out context checking components

and know where to find them later for editing. Here are the issues to address:

1. Add context check definitions to Section XII.

   These complex definitions should be copied from existing sources and pasted to avoid errors. We supply several common context check definitions on the disk included with this book. They are in a file named "Basic Context Check Definitions."

2. Modify task-oriented sequences to include any combination of <Skip If> and/or <Make Sure> context checks.

   To maximize the ease of creating sequences and panels with Guide Starter, it is best to use Guide Starter to generate the longest possible sequence for any task, whereby you assume nothing about the state of the application. Then build Skip Ifs in front of panels that may be passed over if certain conditions are met.

3. When you add a <Skip If> or <Make Sure> to a sequence, you must then define a new sequence in Section VI to which Guide execution branches if the context check fails.

   This kind of sequence, for either Oops or Automated Scripting styles, is a self-contained unit that includes definitions for both the sequence and panel. The panel definition within the unit also contains specifications for buttons and/or calls to AppleScript scripts.

4. Any AppleScript script must be written, debugged, and tested with an external script editor and saved as a compiled script.

   That script's file name is part of the call to the script in the sequence. The Guide compiler blends the script with the Guide text file, so the script isn't needed after the Guide is successfully compiled. For your own edification, we recommend listing the names and purposes of each AppleScript script as commented lines in Section X of the Guide text file.

5. If your Oops or AppleScript sequence panels require any special prompts, you must define appropriate prompt sets in Section III of the Guide text file.

Context Checks for a File Dynamo Guide

You can experiment with context checking by adding checks to a File Dynamo Guide. The context you can test for is whether the Selection Criteria window is open. File Dynamo was designed so that you can access this window either by clicking the Set button or choosing Selection Criteria from the Edit menu. Whenever the window is open, the Edit menu choice is marked with a check. Therefore you can perform either the IsMenuItemChecked() or the ActiveWindow() context check on this program.

Start by creating a sequence in Guide Starter that describes the steps leading to opening this window and one or more panels that describe working in that window. The sequence you create in Guide Starter should be the most complete path a user could take after the program is started (the program must be started in order for the Guide to appear in the Help menu). The following is how one version of this sequence might look after being created in Guide Starter (plus some comments thrown in for clarity):

```
# Original sequence written by Guide Starter in Section VI
<Define Sequence> "How do I narrow the selection of files?»1","How do
I narrow the selection of files?"
  <Seq Nav Button Set> "Start and Huh"
  <Panel> "2294" # Intro to the process
  <Panel> "2396" # Step 1 to click the Set button (includes coachmark)
  <Panel> "3817" # Tips about working in Selection Criteria window
<End Sequence>
```

To this sequence, we will add a <Make Sure> before the third panel to ensure that the window is open as directed in the second panel. We'll even get fancy by creating a Make Sure➠Automated Scripting context check that leads to a new sequence that in turn calls an AppleScript script to open the window for the user:

```
# Add prompt set definition to Section III
<Define Prompt Set> "Item opened","When this window is open, click
Continue.","When this window is open, click Continue.","When this
window is open, click Continue.","When this window is open, click
Continue."
```

```
# Modified sequence in Section VI
<Define Sequence> "How do I narrow the selection of files?»1","How do
I narrow the selection of files?"
  <Seq Nav Button Set> "Start and Huh"
  <Panel> "2294"
  <Panel> "2396"
   <Make Sure> ActiveWindow("Selection Criteria"), "Auto open:
Selection Criteria seq"
  <Panel> "3817"
<End Sequence>

# New sequence added to Section VI
# to handle <Make Sure> context check when it returns FALSE
<Define Sequence> "Auto open: Selection Criteria seq"
<Seq Nav Button Set> NONE
  <Define Panel> "Auto open: Selection Criteria"
   <Format> "Full"
Please wait a moment. Apple Guide is assisting you by opening the
Selection Criteria window.

   <3D button> 1070,1072,Center,GoBack()
   <On Panel Show> DoAppleScript(":AppleScripts:File Dynamo:Open
Selection Criteria")
   <Panel Prompt> "Item Opened"
  <End Panel>
<End Sequence>

# Comments added to Section X about AppleScript script
# "Open Selection Criteria" — clicks on Set button in main window

# Context check definition added to Section XII
<DCC> "ActiveWindow", 'WIND', FRONT, LONG:0, LONG:0,
LONG:10, LPSTRING
```

To round out the description of this context check, we include the AppleScript script, which you would write and save in the AppleScript Script Editor. The File Dynamo program is written in FaceSpan, an interface builder for AppleScript applications. To perform the action of clicking on the Set button via a script, we use the AppleScript syntax and object naming as defined in the program's scripting dictionary, as follows:

```
tell application "File Dynamo"
    activate
    click window item 9 of window 1 ¬
    at position of window item 9 of window 1
end tell
```

Notice, too, that the script automatically brings File Dynamo to the front (activate), in case a user has accidentally clicked on some other application window. We include this script in a folder called File Dynamo inside an AppleScripts folder, should you wish to try compiling your modified script.

Because the program design also offers a checked menu item as a context checking reference point, we could use that context check method instead of checking for an active window. All you must do for that kind of context check is substitute the ActiveWindow() checks with

```
IsMenuItemChecked("Edit","Selection Criteria")
```

and add the corresponding context check definition to Section XII of the Guide text file.

The final touch for this context check would be to add a <Skip If> command to take into account the possibility that the user has already opened the Selection Criteria window. Staying with the same ActiveWindow() context checking as before, we write a main sequence like this:

```
# Further modified sequence in Section VI
<Define Sequence> "How do I narrow the selection of files?»1","How do
I narrow the selection of files?"
 <Seq Nav Button Set> "Start and Huh"
 <Skip If> ActiveWindow("Selection Criteria")
  <Panel> "2294"
 <Skip If> ActiveWindow("Selection Criteria")
  <Panel> "2396"
  <Make Sure> ActiveWindow("Selection Criteria"), "Auto open:
Selection Criteria seq"
 <Panel> "3817"
<End Sequence>
```

No other changes are necessary because the <Skip If> command uses the same context check as the <Make Sure> command. Now if the user chooses this sequence while the Selection Criteria window is open, she'll see only Panel 3817, the one that has instructions on using the window.

Apple Guide Depth

As the discussions in the last two chapters clearly show, Apple Guide offers help system authors substantial flexibility and depth in designing truly helpful assistance. Certainly not every Guide requires all the bells and whistles described in this part, but it should be a comfort to creative minds that Apple has provided a technology to meet many demands. In our final chapter, we give you an aerial view of the Guide Script language to help you appreciate other aspects of Apple Guide's powers.

EIGHTEEN
GUIDE SCRIPT: AN OVERVIEW

Many times in this book, you've seen examples of the Guide Script command language, the commands that help the compiler program turn your text files into Guide databases. For basic Guide databases, Guide Starter assembles all necessary commands in the text file for you. But, as shown in the seven preceding chapters, inserting a few Guide Script commands can add power to your Guides, for example, presenting decision panels that lead to branches in the instructions, triggering AppleScript scripts, and performing context checks.

The complete vocabulary of the Guide Script language consists of more than a hundred commands, plus dozens of built-in constants and other terms. Some commands you use all the time; others you may never use. Details of the entire dictionary are beyond the scope of this book, but you may be interested in knowing the range of commands in the language. We start, however, with some general comments about the language and the way Guide Script encourages you to think about your Guide text files.

Scripting or Programming?

It has become fashionable to assign the label "scripting language" to any computer language intended for users who don't have formal computer

science training. In the Macintosh universe, HyperCard's language, HyperTalk, is called a scripting language. AppleScript builds the "script" coinage into its very name, as does Guide Script. The goal is to avoid scaring away users who aren't members of the pocket protector crowd, the users who want to automate a process or direct the behavior of their machines.

In truth, scripting is the same as programming: You set down a sequence of commands that the computer eventually follows. Scripting, fortunately, usually implies that the vocabulary is closer to plain, everyday language rather than an assemblage of letters and punctuation that looks as though it were written by an extraterrestrial being. Moreover, the scripting environment usually takes care of the complexities that commercial application programmers have to face. For example, when you define a panel with a simple Guide Script command, the compiler and the rest of the Apple Guide system software take care of inserting the text into the panel, adjusting the size of the panel to fit the text, and drawing the panel on the screen when it is needed—the kinds of nasty things that experienced, professional programmers had to figure out for us.

Another term you may hear regarding Guide Script is that it is a "tagging," or mark-up, language. The concept of tagging goes back to some of the earliest days of electronic document preparation, when computers were WYSINAALWYG—What You See Is Nothing At All Like What You Get. An author would create a document and then insert commands (tags) at numerous points to instruct a printer or computerized typesetting machine to make a heading a particular font, font size, and font weight. A tag before an italicized word switched the printer to the italic font; a corresponding tag after the word returned the printer to regular style.

The creators of Apple Guide and Guide Script envisioned language literate, but not necessarily computer literate, authors creating content for Guides and then marking up the text with Guide Script tags prior to running it through Guide Maker. When you look at the text files generated by Guide Starter, however, you certainly notice that there is more to generating a Guide than just assigning tags to a user manual. So to someone who is familiar with tagging and mark-up languages (SGML, for example), a Guide Script document may appear complex. To a programmer, however, it appears simple. And to an experienced scripter, it looks just right.

Defining Moments

One key to successful Guide Script writing is grasping the sense of objects that the language instills in you. The friendly Apple Guide things we see on the screen—panels, sequences of panels, prompts, and coach marks, to name several—are treated like objects in the text file. Every object has a name, which is assigned by a command that defines the object. Then, whenever that object is needed in your Guide text file—no matter how many times it is needed—you invoke that object by its name. Establishing these defined objects allows the Guide to locate them when summoned by a user's making a selection in the Access window.

To demonstrate this concept, we start the process as an author might (if the author didn't have Apple Guide Starter to help out). Assuming the author already has planned the organization of the Guide, the first task would be to define the panels containing the instructional content. We define the following three simple panels

```
<Define Panel> "Instruction Overview"
This text represents an overview for the instruction in the next panel.
<End Panel>

<Define Panel> "Instruction Step 1"
This text contains details of Step 1.
<End Panel>

<Define Panel> "Instruction Step 2"
This text contains details of Step 2.
<End Panel>
```

whose names are Instruction Overview, Instruction Step 1, and Instruction Step 2. To show these panels in the order we want, we must define a sequence—the heart of the script that Apple Guide follows when a user heads down this path—as follows:

```
<Define Sequence> "How do I do this?"
        <Panel> "Instruction Overview"
        <Panel> "Instruction Step 1"
        <Panel> "Instruction Step 2"
<End Sequence>
```

When a user starts viewing this sequence, Apple Guide takes care of activating the left and right arrow navigation buttons and displaying the panel numbers. Notice, too, that the sequence has a name, the wording that appears in the sequence title display area at the top of the Presentation panel.

The sequence appears on the screen after a user selects a Topic in the Access window. In the Full Access window, the left column consists of Topic Areas, while the right contains Headers (the boldfaced entries) and Topics. Structurally, these components are fashioned like an outline: Click on a Topic Area, and its nested Headers and Topics appear; each Header contains nested Topics. Even the way these items are expressed in Guide Script shows their outline heritage as follows:

```
<Topic Area> "Main Topic"
    <Header> "How do I"
        <Topic> "do this?", "How do I do this?"
        <Topic> "do that?", "How do I do that?"
        <Topic> "do the other thing?", "How do I do the other thing?"
```

Unlike panel and sequence definitions, these items are not balanced with <End> commands. Figure 18-1 shows the Full Access window that displays the results of the previous Topic Area, Header, and Topic definitions.

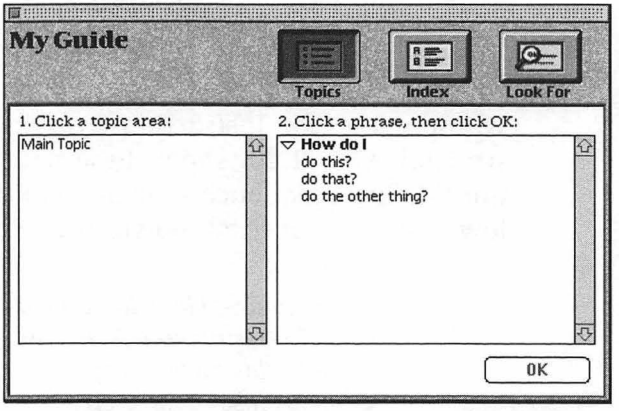


Figure 18-1
A Guide's Access window showing Topic Areas, Headers, and Topics

When a user selects the first Topic, Apple Guide sees that the sequence attached to it is named "How do I do this?" Apple Guide then looks up that sequence and displays the first panel listed in that sequence "Instruction Overview."

More complex panels may call additional items that are defined elsewhere in the text file. For example, a command within a panel definition to draw a coach mark simply invokes that coach mark by name. The coach mark definition contains all the details about the style and location of the coach mark.

If you use Guide Starter to create your sequences and panels, these definitions are all created for you, and you get to think about your Guide in the same "top-down" direction as your Guide's users. But it's important to understand the object orientation of Guide components if you plan to enhance your Guides with the advanced features covered in previous chapters and in *Apple Guide Complete*.

Reusable Components

A helpful byproduct of the definition system in Apple Guide is that once your Guide text file defines an item, that item may be used as often as necessary simply by its being called by name in the appropriate places. In fact, that is what the Index feature of Full Access windows is all about: creating a different interface (alphabetical Index items) to the same sequences and panels that you use in the regular (Topics) portion of your Guide.

Several definitions may be reused beyond a single Guide. If you look at Section II of any text file generated by Guide Starter, you'll see definitions for formats, navigation buttons, and events. These items were extracted from a set of standard definitions provided by Apple Computer (on the CD-ROM that accompanies *Apple Guide Complete*) for common elements, such as the arrangement of text fields in panels. Once you develop a style for your Guides, you can reuse those definitions in all your Guide databases. Guide Script even has a way for you to keep these standard definitions in a separate file that is blended into all of your Guide text files during compilation.

Guide Script Commands

To stimulate your appreciation for the scope of commands in the Guide Script language, we next list the commands (but not their complete syntax or parameters) along with descriptions of what they do. Commands written for you by Guide Starter are marked with an asterisk (*).

Startup Information

<App Creator>	If your Guide file applies to only one application in a folder containing many applications, this instructs only the target application to display the Guide in the Help menu.
<Gestalt>	Instructs your Guide to check values of the Gestalt Manager (*Inside Macintosh: Operating System Utilities*) to see if the Macintosh environment has all hardware and operating system features required for your Guide.
*<Version>**	Inserts information in the version resource of your Guide file for display in the file's Get Info dialog box.
<World Script>	Lets you restrict your Guide database to displaying only on Macintoshes that contain specific writing script and region codes.
*<Help Menu>**	Specifies how the Guide will be listed in the Help menu.
<Balloon Menu Text>	Defines the contents of the balloon that appears when a user points to your Guide's menu item (when Balloon Help is turned on).
*<Comment>**	One of two ways to enter a descriptive line in a text file so that the Guide Maker compiler ignores it (also the # symbol).
<Include>	Blends another text file with the current file prior to compilation.
<Mixin>	Indicates that the Guide file is a Mixin style guide (adds to an existing Guide database).

<Mixin Match>	Lets you assign a code to both a main and Mixin Guide to assure they recognize each other.

Startup Window Specifications

*<Startup Window>**	Defines the type of window the user sees when the Guide comes alive.
*<Howdy>**	Points to the text block definition to be displayed in the Howdy space of an Access window.
*<App Logo>**	Points to a file (for merging into the Guide database) containing logo art for the Access window.
*<App Text>**	In the absence of logo art, the name of the Guide to appear in the Access window.

Default Guide Settings

<Max Height>	Lets you adjust the maximum allowable height of a presentation panel other than Apple Guide's 250 pixel standard.
<Min Height>	Lets you adjust the minimum allowable height of a presentation panel other than Apple Guide's zero pixel standard.
<Default Format>	Sets a defined field format to be the default format for the presentation window.
<Default Nav Button Set>	Sets a defined arrangement of navigation buttons to be the default set for the presentation window.
<Allow Prompts>	Instructs panels to always leave or omit space for prompts at the bottom of panels.
*<Default Prompt Set>**	Sets a defined prompt as the default prompt for the presentation window.
*<Define Prompt Set>**	Lets you specify text for prompt lines in panels, depending on the location of the panel within the sequence (different prompts for first, middle, and last panels, and panels with buttons on them).

Creating Sequences

*<Define Sequence>**	Starts a sequence definition, assigning a name to the sequence.
*<End Sequence>**	Ends a sequence definition.
*<Sequence Prompt Set>**	Points to the defined prompt set to be used for panels in this sequence.
*<Seq Nav Button Set>**	Points to the defined navigation button set to be displayed on panels in this sequence.
*<Panel>**	Points to a defined panel to be used in this sequence.
<Insert Sequence>	Lets a sequence reuse another sequence's panels.
<Jump Sequence>	Allows branching to another sequence.
<Launch New Sequence>	Lets you link one sequence to another when a sequence exceeds Apple Guide's internal limits.
<Build Sequence>	Lets your Guide database include a sequence that doesn't appear in any Access window (used mostly with programs that communicate directly with the database).

Creating Panels

*<Define Panel>**	Starts a panel definition, assigning a name to the panel.
*<End Panel>**	Ends a panel definition.
*<Panel Prompt>**	Call a defined prompt set to override the default prompt set.

Buttons

<Standard Button>	Displays a two-dimensional button (art from a PICT resource) on a panel, including one that invokes a defined event.
<3D Button>	Displays a three-dimensional button (art from a PICT resource) on a panel, including one that invokes a defined event.
<Radio Button>	Displays a radio button on a panel.

<Radio Button ¬ Launch New Seq>	Displays a radio button on a panel and specifies a new sequence to be launched when the user clicks the right arrow with this button turned on.
<Checkbox>	Displays a checkbox on a panel.
*<Define Nav Button>**	Defines a button (art from a PICT resource) to be used in the navigation bar at the bottom of a panel.
*<Dimmable Button Data>**	Displays as active a defined navigation button that is normally dimmed.
*<Define Nav Button Set>**	Establishes a group of defined navigation buttons into a set that can be called by name.

Text Blocks

*<Define Text Block>**	Starts a definition for a chunk of text to be called from other places (for example, Howdy).
*<End Text Block>**	Ends a text block definition.

Text and Object Formatting in Panels

*<Define Format>**	Define specifications for a text field and related objects in a panel.
<Define Transparent Format>	Define specifications for an overlapping, transparent text field.
*<Format>**	Specify a defined field format for a chunk of text in a panel.

Pictures and Movies in Panels

<PICT>	Insert a picture from a PICT resource into a panel.
<QuickTime>	Insert a movie from a QuickTime movie file into a panel.

Resources

*<Resource>**	Instructs the compiler to blend resources from a resource file into the Guide database for use by panels or sequences.
<Starting Res Number>	Manages resource IDs in the Guide database.

Coach Marks

*<Define Menu Coach>**	Defines specifications for a single menu coach mark.
<Define Item Coach>	Defines specifications for a single dialog box coach mark.
<Define Object Coach>	Defines specifications for a coach mark guided by interaction with a specially written application.
*<Define Window Coach>**	Defines specifications for a single window coach mark.
<Define AppleScript Coach>	Defines a coach mark based on coordinates returned from an external AppleScript script.
*<Coach Mark>**	Displays a defined coach mark in a panel.

Hot Items

<Hot Object>	Assigns event to be carried out when a user clicks the object or text named in the next line of the panel definition.
<Hot Rectangle>	Defines the rectangular area of a panel that acts as a hot object and the event to be carried out.
<Hot Text>	Defines the text in the next text block that is hot and the event to be carried out.

Topic Areas

<Topic Areas Instruction>	Overrides the label above the Topic Areas field in a Full Access window.
*<Topic Area>**	Assigns text to a Topic Area entry in a Full Access window.

Index Terms

<Index Instruction>	Overrides the label above the list of indexed items in a Full Access window.
*<Index>**	Assigns text to an Index entry in a Full Access window.
<Sorting>	Specifies sorting method (international or U.S. ASCII) of Index entries in a Full Access window.

<Index Sorting>	Specifies whether Apple Guide sorts Index entries based on displayed text or hidden keys (defined by <Index> command).

Headers and Topics

<Topics Instruction>	Overrides the label above the list of Topics and Headers in Full and Single-list Access windows.
*<Header>**	Assigns text to a boldfaced header.
*<Topic>**	Assigns text to a Topic entry.

"Look For" Help

<Look For Instruction>	Overrides the label above the search phrase entry box in the Full Access window.
<Look For String>	Sets default text appearing in the search phrase entry box.
<Ignore>	Indicates a word to be ignored from the search phrase entry box prior to performing the search.
<Exception>	Specifies a search phrase entry word that should not be stemmed (truncated) to its root word by removing plural, participle, gerund, and other endings.
<Synonym>	Specifies a synonym that can be accepted for a true index entry.

Conditional Commands

<If>	Start of a conditional expression, followed by command(s) to be executed if the condition returns true.
<Else>	Specifies command(s) to be executed if a previous <If> condition returns false.
<End If>	Required ending for a conditional construction.
<Skip If>	Specifies a condition to test, instructing Apple Guide to pass over the next command if the condition returns true.
<Make Sure>	Specifies a condition to test and a sequence to branch to if the condition returns false.

<Start Making Sure>	Specifies a condition to test prior to displaying any subsequent panel, until the next <End Making Sure> command, plus a sequence to branch to if the condition returns false for any panel.
<End Making Sure>	Required ending for a <Start Making Sure> command.

Context Checks

<Define Context Check>	Defines parameters and external module resource for a context check.

Events

*<Define Event>**	Defines an event to execute for button clicking or other user actions that can trigger events.
<Define Event List>	Defines a series of events to execute in response to button clicking or other user actions that can trigger events.
<On Panel Create>	Specifies a defined event to execute prior to a panel displaying itself.
<On Panel Destroy>	Specifies a defined event to execute immediately after a panel disappears.
<On Panel Show>	Specifies a defined event to execute while a panel displays itself.
<On Panel Hide>	Specifies a defined event to execute while a panel disappears.

Mixin Guide Commands

<Replace Sequence>	Substitutes a sequence in the current Guide for an existing sequence in the main Guide.
<Insert Topic Area Header>	Links a new Header to a Topic Area in an existing Guide.
<Insert Topic Area Topic>	Links a new Topic to a Topic Area in an existing Guide.
<Insert Index Header>	Links a new Header to an Index entry in an existing Guide.
<Insert Index Topic>	Links a new Topic to an Index entry in an existing Guide.

<Delete Topic Area>	Specifies a Topic Area in an existing Guide to be hidden.
<Delete Topic Area Header>	Specifies a Header in an existing Guide to be hidden.
<Delete Topic Area Topic>	Specifies a Topic in an existing Guide to be hidden.
<Delete Index>	Specifies an Index entry in an existing Guide to be hidden.
<Delete Index Header>	Specifies a Header in an existing Guide's Index view to be hidden.
<Delete Index Topic>	Specifies a Topic in an existing Guide's Index view to be hidden.

Command Abbreviations

While Guide Starter fully spells out all Guide Script commands that it generates, you may see Guide Script documents whose commands use much shorter abbreviations. Guide compilers accept these abbreviations, which are most often the first letters of multiple-word commands. For example, <Define Context Check> can also be written as <DCC>. Most one-word commands do not have abbreviations.

Learning Guide Script

The best way to learn any programming language is to study examples. For Guide Script, a good way for you to begin this study is to examine the text files created by Guide Starter. You are familiar with the content and the way panels and sequences interact with each other in the Apple Guide interface. By studying the text files, you should begin to understand the way basic commands work.

The next step is to begin experimenting with some of the "bells and whistles" described in Chapters 11 through 17. Begin making simple modifications to the Guide Starter text file and recompiling the file. For a more thorough understanding of Guide Script commands and all the possibilities, however, we recommend moving up to *Apple Guide Complete*. The CD that accompanies that book contains the source text files for Macintosh Guide, the large database that comes with System 7.5, as well as additional system software Guides. You'll learn there, for example, more ways to structure a very large Guide text file.

We still encourage you to come back to Guide Starter for all your Apple Guide projects. You can let Guide Starter handle the drudgery of setting up the organization of your entire Guide and defining panels and sequences. Then you can use the text file to make whatever tweaks and additions are needed for advanced features.

We hope you find Guide Starter and the information in this book helpful in getting your Apple Guide efforts going quickly and smoothly. And we look forward to hearing about the great Guides you are creating for others in the Macintosh community.

INDEX

Addison-Wesley warrants the enclosed disk to be free of defects in materials and faulty workmanship under normal use for a period of ninety days after purchase. If a defect is discovered in the disk during this warranty period, a replacement disk can be obtained at no charge by sending the defective disk, postage prepaid, with proof of purchase to:

Addison-Wesley Publishing Company
Editorial Department
Trade Computer Books Division
One Jacob Way
Reading, MA 01867

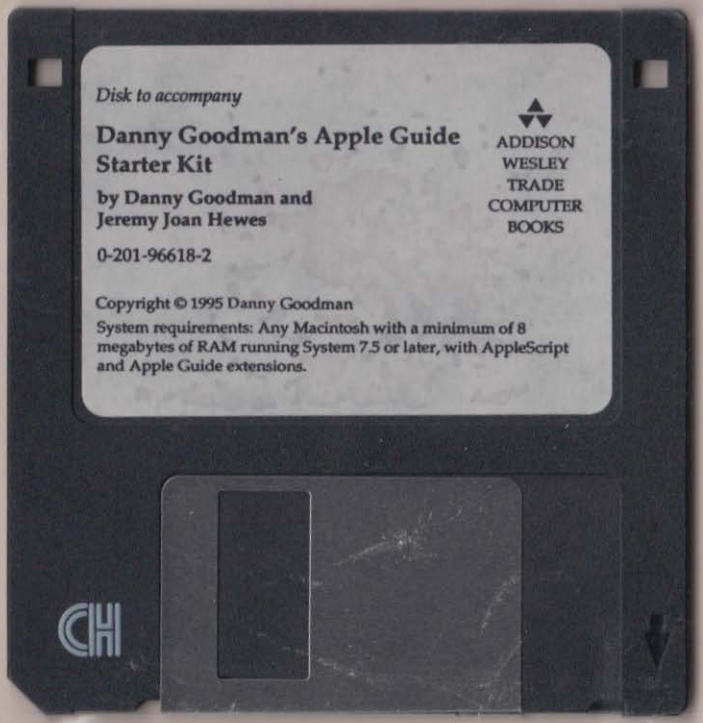
After the ninety-day period, a replacement will be sent upon receipt of the defective disk and a check or money order for $10.00, payable to Addison-Wesley Publishing Company.

Addison-Wesley makes no warranty or representation, either express or implied, with respect to this software, its quality, performance, merchantability, or fitness for a particular purpose. In no event will Addison-Wesley, its distributors, or dealers be liable for direct, indirect, special, incidental, or consequential damages arising out of the use or inability to use the software. The exclusion of implied warranties is not permitted in some states. Therefore, the above exclusion may not apply to you. This warranty provides you with specific legal rights. There may be other rights that you may have that vary from state to state.

Danny Goodman and Jeremy Hewes are available to answer your Apple GuideStarter Kit questions on the following online forums:

- eWorld: Use the keyword "MACDEV"; double-click the Technical Discussions buttons, and open the Apple Guide folder.
- America Online: Use the keyword "APPLESCRIPT" to reach a forum where Apple
Guide authors hang out.
- CompuServe: Use the keyword "MACDEV" to reach the Macintosh Developers forum.
- Apple Guide Internet mailing list: send an e-mail message to listserv@list.peter.com.au with the text "subscribe apple-guide <Your Rea Name>" in the message body.

Forum organizations are subject to change. The above information was accurate a this book went to press.


Disk to accompany
Danny Goodman's Apple Guide Starter Kit
by Danny Goodman and Jeremy Joan Hewes
0-201-96618-2
ADDISON WESLEY TRADE COMPUTER BOOKS
Copyright © 1995 Danny Goodman
System requirements: Any Macintosh with a minimum of 8 megabytes of RAM running System 7.5 or later, with AppleScript and Apple Guide extensions.

"*Danny Goodman's Apple Guide Starter Kit* software makes it incredibly easy for anyone to get started writing Guide files."
—GLENN KATZ, guideWorks LLC, Apple Guide Designer

Apple Guide is the sophisticated, interactive online help system provided with Macintosh® System 7.5 *Danny Goodman's Apple Guide Starter Kit* provides you with an automated tool and quick, hands-o instructions for creating your own Apple Guide databases for any task or procedure with the Macintosh

The *Starter Kit* includes everything you need to write and compile simple, effective Guides. Using th Apple Guide Starter program included on the disk, you can make Guides quickly and easily, withou having to learn a scripting language or write coded files (which is what you'd have to do without the pro gram). The authors provide expert advice on how to design a good Guide, from planning and creatio through testing, revising, and indexing. And you'll get tips on how to enhance your Guides by addin graphics, decision panels, context checks, and more.

Some of the ways you can put the program to immediate use include making Guides for:

- tasks that involve more than one program, and thus would require you (or your coworkers or clients) to look in several places for complete instructions
- any number of customized procedures for a business, such as preparing invoices, updating archives or inventories, connecting to an online information service, or searching a large database
- procedures for keeping sensitive information secure and for backing up important data
- a variety of school uses, such as instructions for classroom projects, computer lab assignments, quizzes, and computer training for teachers and students.

For novice and experienced users alike, ***Danny Goodman's Apple Guide Starter Kit*** provides an all-in-one, easy-to-use package for creating interactive instruction.

DANNY GOODMAN is a freelance writer, Macintosh industry watcher, and author of more than twenty books, including the *Complete HyperCard Handbook*, now in its fourth edition, and the *Complete AppleScript Handbook*.

JEREMY JOAN HEWES has spent the past eight years with Apple's Instructional Products group, and has worked with the Apple Guide team from its inception. She is the author or coauthor of six books, including *Writing in the Computer Age*.

53495
9 780201 483499
ISBN 0-201-48349-1
$34.95 US
$48.00 CANADA

Cover design by Jean Seal

Addison-Wesley Publishing Company