3

Planning the JavaHelp Project

The secret of any successful project, whether it's writing online help or building a house, is good planning. Imagine you want to drive across the country, and you decide to just jump in the car and start driving. Unless you've made the same trip several times in the recent past, you probably won't reach your destination without encountering major problems.

Developing online help is no different. Unless you have a solid plan for your project, you will likely encounter many problems. These problems can add up and negatively affect your final online help system. Project planning helps ensure that your JavaHelp project runs smoothly and that you end up with a usable online help system. It won't remove all obstacles, but good project planning will help you avoid unforeseen problems.

This chapter is about JavaHelp project planning. If you are an experienced help author, you may already have a project-planning system that works well for you. If so, you may want to skim through the chapter to look for sections that are specific to JavaHelp development. If you are new to help-authoring, you should read the entire chapter. It provides a general explanation of project planning by discussing the following topics:

- General planning tasks
- Planning tasks specific to JavaHelp

TIP When planning your project, consider using a third-party help-authoring tool to make help-authoring easier. Third-party tools automate many of the tasks for JavaHelp that would otherwise be tedious and time-consuming. Chapter 9, *Using Third-Party Help-Authoring Tools*, covers JavaHelp-authoring tools and explains why you should understand how JavaHelp works even when using a help-authoring tool to create a HelpSet.

General Planning Tasks

Project planning is highly subjective. If you ask several help authors how to plan a project, you'll probably get several different answers. These differences can be both good and bad. Differences in opinion and new ideas are what drive theory and technology forward. However, for a new help author trying to get some direction on how to plan a new online help project, mixed opinions could only complicate matters. In the following sections, I compare several project-planning theories. My intent is not to claim that other theories are wrong but instead to help you determine a good practice for yourself.

Project planning is quite involved—so involved that authors write entire books on the subject. Since this book is about developing JavaHelp projects, I discuss only project-planning topics I think will help you develop a solid JavaHelp project. If you find a need or desire to learn more about project planning, I encourage you to pick up a book on the subject. In this section, I discuss some general concepts of project planning for online help systems.

Defining the Audience

Before you can outline or write online help topics, you must know the audience of the help system. Start your project by defining your audience. Doing so will help you determine the content of your online help topics, the audience's experience level, any additional elements you should include (such as images or multimedia), the way the audience should access the help system, and any special considerations for navigation. Careful audience definition will help you accurately perform the rest of the project planning and online help development tasks.

NOTE In most cases, defining the audience of an online help system essentially means defining the users of the software for which the help is written. For this reason, I use the words *audience* and *users* interchangeably.

There are many different ways to learn about your audience. The important thing is to use all possible resources to find as much information about them as possible. Depending on the resources you have available, you should consider the following options for learning about your audience:

Study user information based on previous releases of similar software or from competitors' software

If you are writing or updating online help for a software upgrade, use audience information from the previous release such as interviews, surveys, customer-support calls, or any other source of user feedback. If your application is new, you can look at online help from competitors' software, which may help you understand the types of users who may work with your application.

Talk with customer or technical support personnel, who interact with users on a regular basis

Customer or technical support personnel frequently talk with users who generally call in with problems or other feedback. Use their information and incorporate it into your online help systems. In the long run, you'll increase user satisfaction and help cut down on support calls.

Talk with marketing personnel, who may have already conducted studies on their users

Marketing personnel should know everything about the people buying their software. Talk with them to get information based on research they may have performed to market the product.

If you are a technical writer, talk with software developers who may have conducted user studies of their own

Software developers are in the same situation as you. They must study the potential users of their software to know how they should develop the application. Sharing information with them provides shortcuts and saves time for the entire development team.

Talk with trainers who may have taught classes for the users of your software

Because of the nature of classroom interaction, trainers probably know the questions most frequently asked by your users. Talk with them and generate a list of frequently asked questions. Use this information when planning the topics for your HelpSet.

Assess the nature of the software

Look at the software itself to define the audience. For example, if the software is a sophisticated data-warehousing tool, you know your audience consists of database programmers and other technical people. On the other hand, if the software is an S.A.T. preparation course, your audience consists of high school students who are preparing for college. The language you use for these two audience types and the experience level you assume are completely different from one another.

,ch03.7524 Page 31 Thursday, April 20, 2000 12:36 PM

Determining the Audience's Needs

Once you have determined your audience, you can define their needs. There are many aspects to consider when defining the audience's needs. You must consider every situation your users are in as they use the software and access its online help. Consider the following factors as you define your audience's needs:

Input from users

Some companies perform complete needs analyses by talking with potential users of their application. If you are part of such a company, you can ask those users what they need and want in an online help system.

Need for context-sensitive and embedded help

Depending on the users of your software, context-sensitive and embedded help can be a great service. People with little computer experience would welcome such help because it reduces their workload. With context-sensitive help, the system displays the appropriate information based on the application's current situation. Embedded help makes using online help easy since it's always displayed. Context-sensitive and embedded help are usually welcomed by novice users, who need a lot of assistance. However, expert users might become a bit annoyed with embedded help if the application is always forcing online help on them.

Experience level of users

I just mentioned one example of considering different users' experience levels. There are, however, many more considerations. The style you use to write every help topic depends on the experience level of your users. Experienced users are comfortable with technical terms, while novice users need concepts presented in simple terms. Be sure to gear your information toward the right user level. Always plan for the "lowest common denominator." If you have a mix of users, make sure you write for the lowest experience level and use an intuitive navigation design so that users of all experience levels can find the appropriate information.

Age of users

You should consider the age of your users when writing your help topics. The language and tone you use in your writing will be different for a teenager than for an elderly person. The way you come across to your audience determines the credibility you establish with them.

Situation under which users work with the application

The situations under which users work with your application will dictate a large amount of their needs. For example, if the application supports work performed in an office or other business setting, users are probably rushed while working with the application. If users work with the application at home, they are probably not as rushed. You could probably imagine the difference in the way you would have to deliver online help for an application that helps air traffic controllers land aircraft compared to an application that helps people plan a family vacation.

Any disabilities the user may have

Don't overlook the possibility of disabilities your users might have. Simple tasks, such as moving the mouse and clicking a button, can be difficult for people with certain physical disabilities. If your software is designed to assist people with physical disabilities, you should consider how embedded and context-sensitive help could work with the application. Embedded and context-sensitive help can relieve the user of having to frequently access help and search for specific topics.

Planning Integration with Hardcopy Documentation

A misconception among some help authors is that you can simply convert hardcopy documentation to electronic documentation, throw in a TOC and index, and then end up with a usable online help system. This practice does little to help your audience. The way people use online help is very different from the way they use hardcopy documentation. You therefore must consider how your online help will integrate with any hardcopy documentation included with your application.

A common-sense approach is to think about how you read documentation yourself. When you first purchase software and want to read overview material to understand the basic concepts of how the application works, you probably don't want to run the online help and start reading a lengthy explanation on the screen. Instead, you probably would rather open up a book, grab a comfortable seat, and read about the basics of the application. Conversely, if you are in the middle of a procedure with the application and quickly need to refer to instructions, you don't want to interrupt your workflow to go grab a book off the shelf. Instead, you want to click a button that launches the online help and view the instructions while working with the application.

When planning the content of your online help system, you should determine if certain material should be presented online or if it would be better presented in a book. In general, the following types of information work well in online help:

- Short, general concepts that help users better understand the tasks they are trying to perform
- Procedures for tasks
- Brief reference material
- Troubleshooting information for performing tasks

Notice that for troubleshooting information I said *for performing tasks*. Troubleshooting for system problems (as opposed to task-related problems) should be placed in a hardcopy manual. If users have problems and can't run the software application, they might not be able to access the application's online help. System-related troubleshooting information does little good in a help system if users can't access it.

You should also consider whether or not you will be providing context-sensitive help. There is no need to explain the application's windows and controls in hardcopy documentation if you will be doing so with field-level help. Users want to know what controls do when they are thinking of using them—not when they are trying to understand the basics of the application.

One trick I use when deciding how to integrate the online help with hardcopy documentation is to picture the user reading the online help while working with the application and trying to accomplish a task. I then picture the same user reading nonreference hardcopy documents (such as a user's guide or getting-started book) away from the computer. From these models, I decide what information users need in each situation and place the information appropriately.

Creating an Outline

Before you put your online help system together, create an outline. The outline will be the blueprint for your online help system. A good outline should include every aspect of your help system including types of help topics, content, navigational structure, images, multimedia, and other enhancements you need for your system.

During the project-planning stage, you probably won't know everything about the application for which you are writing help. For this reason, you can't expect your outline to include a list of every help topic in the final HelpSet. Instead, come up with general subjects from which you can develop the final help topics. You should also consider the need for images, multimedia, and other enhancements. You will need to have an idea of the quantity of these enhancements so that you can estimate their development time when you create a schedule.

One important aspect to consider is how the users will access the help system. If the help system supports a web-based applet, and users will be connecting to the Web with a modem, you have to plan a system with smaller files. Using a modem, your audience won't want to wait for larger files to download. This demand requires you to minimize the use of larger files such as images and multimedia. If your users access the applet through a local network, you can afford to use larger files since there is usually little download time over local networks.

Another consideration is whether the users install the application directly on their computer. In this situation, you need to consider the size of your files, not because of their download time, but because you don't want a lot of space taken up by help files. If you plan on using multimedia, consider storing the multimedia files on a CD-ROM and accessing them from the CD-ROM as needed.

Of course, these scenarios reflect only a fraction of the possible needs your users may have. The bottom line is that your outline must account for the needs you determine in the early planning stages. The best way to demonstrate the outline is through an example.

Example of a general outline

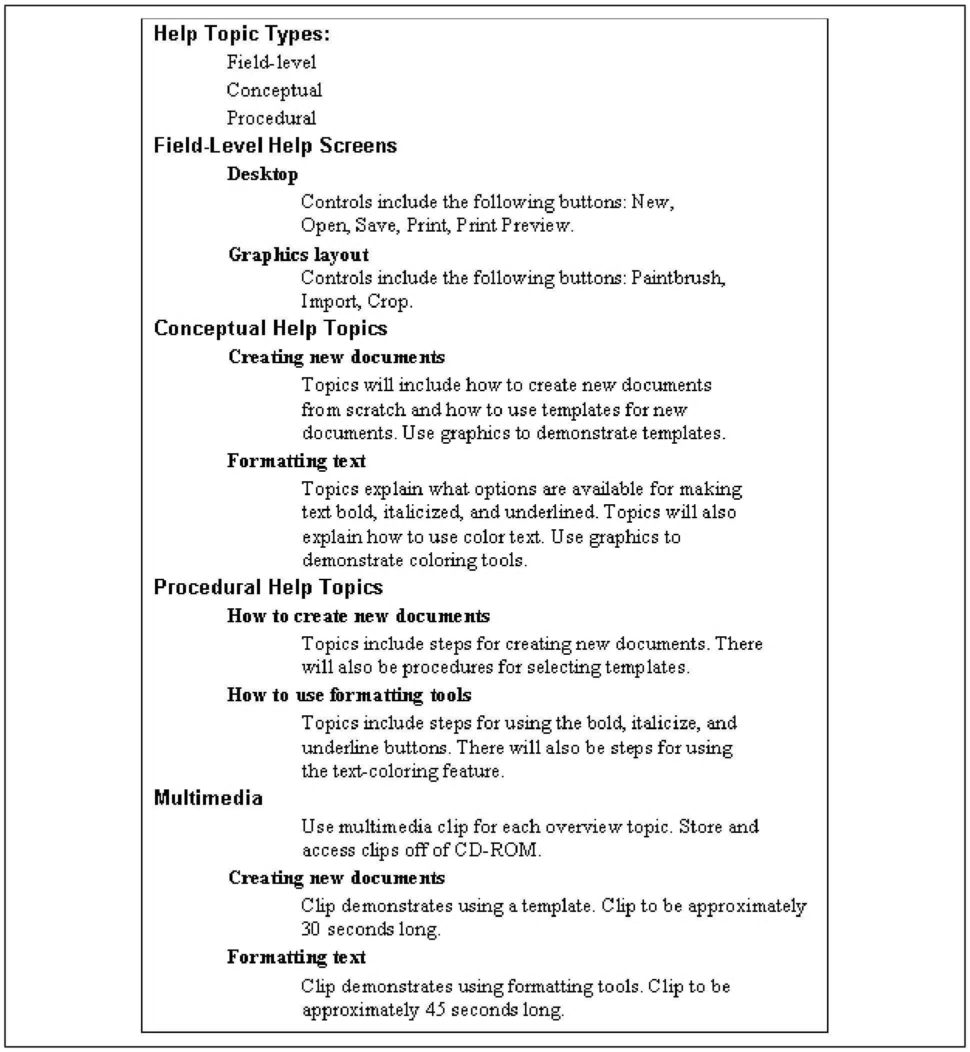
Suppose you want to create an outline for the online help system of a word processing application. You would first outline the types of help topics, along with their general content. You would probably decide to have field-level help for every control on every screen in the application. This type of help would contain only brief descriptions of the controls' functions. You would also want to estimate how many controls are needed for field-level help. This knowledge will help you later when you schedule your project tasks.

You also should have conceptual and procedural help—the type of help users see when they click a **Help** button. You would probably decide that you want conceptual and procedural help for topics such as creating new documents, formatting text, using editing tools, incorporating images, as well as other topics related to the word-processing application. Again, you should try to estimate the number of help topics you will need to schedule the project tasks.

The application will run directly on the user's computer, so you don't have to worry about download time for images. You might decide to use multimedia clips since you found that many novice users new to word processing will be using the application. To provide multimedia, you will want to plan on incorporating the clips on a CD-ROM. You will probably end up with an outline similar to the one shown in Figure 3-1. In reality, of course, your outline would be much longer since you would have more topics and longer descriptions.

Testing Your Outline

Once you have an outline, you should test it. There are many ways to test your outline depending on available time and resources. If you have minimal time and resources, a test can be as simple as giving the outline to potential users to see if the proposed system answers their questions. If you have ample time and resources, the test could mean setting up a role-playing environment where users



Help Topic Types:

Field-level

Conceptual

Procedural

Field-Level Help Screens

Desktop

Controls include the following buttons: New, Open, Save, Print, Print Preview.

Graphics layout

Controls include the following buttons: Paintbrush, Import, Crop.

Conceptual Help Topics

Creating new documents

Topics will include how to create new documents from scratch and how to use templates for new documents. Use graphics to demonstrate templates.

Formatting text

Topics explain what options are available for making text bold, italicized, and underlined. Topics will also explain how to use color text. Use graphics to demonstrate coloring tools.

Procedural Help Topics

How to create new documents

Topics include steps for creating new documents. There will also be procedures for selecting templates.

How to use formatting tools

Topics include steps for using the bold, italicize, and underline buttons. There will also be steps for using the text-coloring feature.

Multimedia

Use multimedia clip for each overview topic. Store and access clips off of CD-ROM.

Creating new documents

Clip demonstrates using a template. Clip to be approximately 30 seconds long.

Formatting text

Clip demonstrates using formatting tools. Clip to be approximately 45 seconds long.

Figure 3-1. A help project outline

simulate working with the application while you simulate providing online help based on the topics in your outline.

Whatever means you use to test the outline, your goal is the same. By testing your outline, you are making sure it covers the information needed by your audience. If you find there are holes in your outline, you can go back and fix the outline before you use it to schedule tasks.

Scheduling Project Tasks

As with project planning in general, different help authors have different opinions on how to schedule tasks. Some help authors prefer to schedule the entire project at the beginning of project planning. Their theory is that, by scheduling at the beginning of project planning, you can add in the time necessary for planning the project itself. You will have to find a style that works best for you, but I don't think you can accurately schedule a project until you know the topics and features the online help system will contain. You won't know this information until the end of your project planning. I therefore present information based on scheduling tasks at the end of project planning.

Different help authors also have different opinions on how you should allocate time for tasks. I've heard some theories that state you should take the total time for all of the tasks and multiply it by a number, such as 1.5 or 2, to account for unknowns that may come up during the project. Again, you have to find a system that works for you, but I don't believe in multiplying times by arbitrary numbers. Instead, I believe you should predetermine the different setbacks you might face during the project and account for them in your schedule. In general, you should consider the following issues when setting a task schedule:

Time required to write help topics, edit help topics, create images, develop multimedia files, and create any other supporting files

These tasks should be your largest consideration. They make up the bulk of your project and therefore dictate, for the most part, how long the project will take. Many help authors use a formula to determine the amount of time it takes to research, write, and edit each topic in a HelpSet. They usually calculate approximately four hours per conceptual or procedural help topic and approximately one hour per field-level help topic. Using a formula like this is acceptable to calculate an estimate when you have no other information with which to plan, but you must remember that this is only an estimate.

The more intimate you are with your software and the more aware you are of the tasks required to develop the HelpSet (such as creating images and multimedia), the more accurately you can calculate the number of hours for each help component.

Number of personnel working on the online help project and their experience level

Your project may or may not go more quickly if you have more people working on it. You also must consider the experience level of the people working on the project. Having experts for each type of component (such as images and multimedia) increases the speed of development. On the other hand, using interns or entry-level personnel can decrease the speed of development.

Time required to learn new software

If you plan to use new software to write help topics, create images, create multimedia clips, or perform other tasks, you must keep in mind the time spent to learn the new software. As you become familiar with JavaHelp, you will notice an improvement in development time. You also should consider the time spent to troubleshoot problems or contact customer support if you must resolve issues with the new software.

Time to test the finished online help system

You must build in time for testing the online help system and its components. The amount of time you estimate for this task depends on your resources. Some companies have quality-assurance personnel who are experts at testing software. Other companies require help authors to test their own systems. When you plan for testing, be sure to account for all components of the help system, such as the navigation components and multimedia clips. You will probably be more successful if you schedule testing concurrently with development.

Time to integrate the online help system with the software application

Once you have a functioning help system, you have to connect it to the software application. This process can be time-consuming; it depends on the presentation option you choose. If you include a single button to launch a standalone Help Viewer, it won't take much time to make the connection. However, when developing context-sensitive help, you must include the time it takes to assign each help topic to a specific object within the application.

Deadlines set by project leaders or other team members

Often, help authors are not allocated enough time to develop a good help system. Project deadlines and other project-related constraints pose certain obstacles to scheduling. In such cases, you must determine which tasks you are willing to sacrifice and settle for whatever help system you can best create in the limited time.

Planning Tasks Specific to JavaHelp

In addition to general project planning concepts, you must focus on tasks specific to JavaHelp development. Because of the function and features of JavaHelp, you must also consider several factors when planning your JavaHelp project.

Deciding How to Present the HelpSet

The way in which a HelpSet is presented to users—as standalone, context-sensitive, or embedded help—dictates how much extra time you need for development. During most of the development cycle, you'll access the HelpSet in a standalone manner. If your goal is to have a standalone HelpSet, you need only a mechanism for launching the JavaHelp system. In many situations, however, you have to improve usability by presenting the HelpSet as context-sensitive or embedded help.

For context-sensitive help, you must account for the time involved in assigning map IDs (from the map file) to the appropriate controls in the application. The amount of time to budget depends on how you connect the context-sensitive help. If you only connect context-sensitivity for the application's windows, it shouldn't take too much time to assign map IDs. However, if you are providing field-level help and must assign map IDs for every object and every window, you should study the application's interface to estimate the number of objects to be assigned to map IDs.

If you are a technical writer, implementing embedded help requires close coordination with the application developer because embedded help is part of the application's interface. Discuss embedded help early during project planning, so the developer can plan for it.

Deciding How to Install the HelpSet

When you plan your JavaHelp project, you must consider how the HelpSet will be installed at the user's site. In Chapter 1, *Understanding JavaHelp*, I discussed different JavaHelp installation methods. It's important to decide on a method early enough to plan for it. You should also consider how your decision might affect the use of enhancements in your HelpSet. Earlier in this chapter, I discussed considerations for enhancing your HelpSet with images or multimedia. You must be careful with such enhancements because you don't want to frustrate your audience by making them wait for file downloads. In general, use the following guidelines for incorporating images and multimedia:

- For an independent Java application, images work well, but you should put multimedia files (depending on the number of files and their size) on a CD-ROM.
- For applets or applications that run on a local network, include images and multimedia clips on the network server.
- For applets that run in a web browser (assuming users connect with a modem), minimize the use of images. Multimedia clips are generally not feasible unless you can use *streaming*, a multimedia technology in which users can start viewing a multimedia clip before the file has completely downloaded to their computer.

Outlining the File and Directory Structure

The preceding chapter included a brief presentation of a HelpSet's file and directory structure. Since topic-to-topic links depend upon the correctness of relative URLs, it is worth outlining the entire directory and file structure before writing individual help topics. Keep in mind that third-party help-authoring tools (discussed in Chapter 9) can help you organize your project directory and file structure.

The first directory you must create is the main project directory, in which you place all other project directories and files. Name this directory after the application for which the help is written. For example, for the sample word processor application used earlier in this chapter, the main directory is named *WordProcessorHelp*. Figure 3-2 shows the directory structure for the *WordProcessorHelp* project.

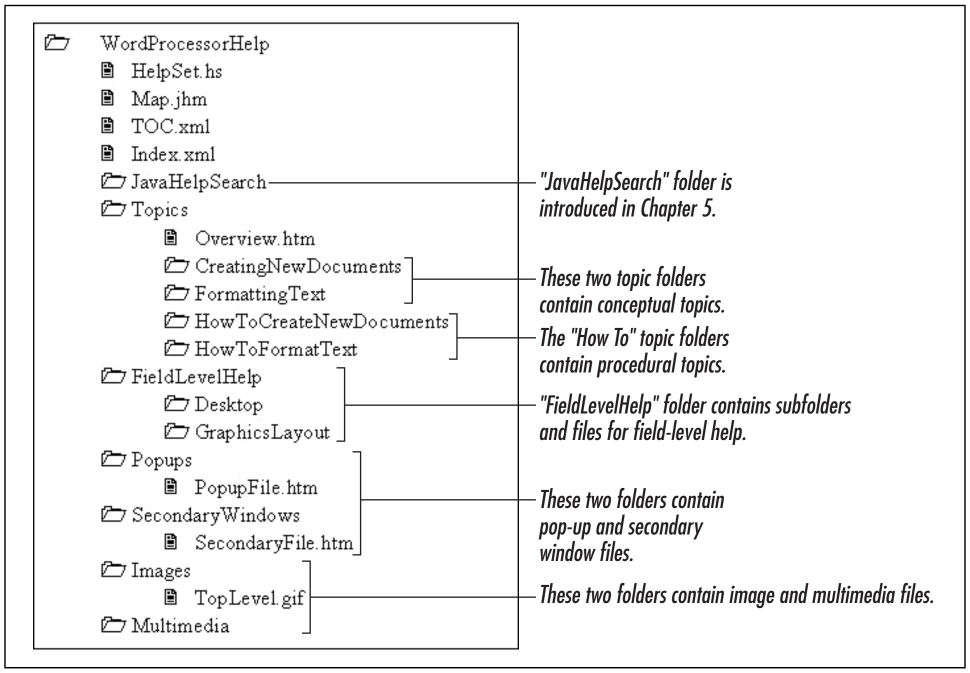

WordProcessorHelp
HelpSet.hs
Map.jhm
TOC.xml
Index.xml
JavaHelpSearch
Topics
Overview.htm
CreatingNewDocuments
FormattingText
HowToCreateNewDocuments
HowToFormatText
FieldLevelHelp
Desktop
GraphicsLayout
Popups
PopupFile.htm
SecondaryWindows
SecondaryFile.htm
Images
TopLevel.gif
Multimedia
"JavaHelpSearch" folder is introduced in Chapter 5.
These two topic folders contain conceptual topics.
The "How To" topic folders contain procedural topics.
"FieldLevelHelp" folder contains subfolders and files for field-level help.
These two folders contain pop-up and secondary window files.
These two folders contain image and multimedia files.

Figure 3-2. Directory and file structure for a JavaHelp project

You might use a modified directory structure if you plan to merge different HelpSets. These considerations are discussed with the information on merging HelpSets in Chapter 6, *Enhancing the HelpSet*.

Structuring HelpSet data and navigation files

When you outline your file and directory structure, keep in mind that all JavaHelp files refer to each other by their relative positions to one another:

- The HelpSet file refers to the map file and the navigation files (TOC, index, and word-search index).
- The map file refers to all the help topic HTML files.

For example, a HelpSet file might refer to the map file like this:

```
<mapref location="Map.jhm"/>
```

The filename *Map.jhm*, with no directory location, indicates that the map file is in the same directory as the HelpSet file.

I strongly suggest that you keep the HelpSet, map, and navigation files in the main project directory. The files reference each other by filename only, without having to specify any directories.

Structuring topic and field-level help directories and files

All topic directories and files are listed relative to the location of the map file. For example, the map file from the preceding chapter includes these lines:

```
<mapID target="computers" url="Topics/Interests/Computers.htm"/>
<mapID target="fitness" url="Topics/Interests/Fitness.htm"/>
```

The directory *Topics*, along with the subdirectory *Interests* and file *Computers.htm*, is listed relative to the map file: the *Topics* directory is at the same directory level as the map file.

I strongly recommend having one major *Topics* directory at the same level as the map file, as illustrated in Figure 3-2. Notice that I placed the *FieldLevelHelp* directory outside the *Topics* directory. You usually have the word-search index utility search the entire *Topics* directory to build its search database. (I discuss this utility in Chapter 5, *Creating HelpSet Data and Navigation Files*.) Excluding the field-level help files from that directory means they won't show up in any word search. If you want to include field-level help in the word-search index, place the *FieldLevelHelp* directory under the *Topics* directory. (Alternatively, add the topics to the search database manually, as discussed in Chapter 6.)

Structuring images and multimedia files

The main project directory should also have directories to hold any images and multimedia files. Figure 3-2 show *Images* and *Multimedia* directories. As with each of the topic directories, use these directories to store files as you develop them throughout the project.

Structuring pop-up and secondary window files

A final consideration when outlining your file and directory structure is whether or not you want the JavaHelp system to display some your HelpSet's topics in pop-up windows and secondary windows. As discussed in Chapter 6, pop-up and secondary window topics are simply HTML files, just like the rest of your topic files. However, you should consider storing these files outside your *Topics* directory for two reasons:

- You will find it easier to stay organized if you keep pop-up and secondary window files in their own directories.
- You will probably want to exclude pop-up and secondary window topics from the word-search index.

Therefore, as with the *FieldLevelHelp* directory, you should place these directories outside of the *Topics* directory.

Planning the Navigation Components

Planning for your navigation facility depends on the size of your HelpSet and whether or not you use a third-party help-authoring tool to create it. If you use a third-party tool, the time to create the navigation facility is negligible—since the third-party tool does the work for you. If you create your HelpSet manually, you should plan on creating the navigation components as you write the online help topics. Add enough time to your schedule to add each topic to the TOC and index, keeping in mind that there will probably be several index entries for each topic. You shouldn't have to plan too much time for the word-search index, since JavaHelp includes a utility for generating it.