,ch02.29521 Page 18 Thursday, April 20, 2000 10:54 AM

2

Creating Your First HelpSet

Now that you have a basic understanding of how JavaHelp works, it's time to build a small HelpSet. The purpose of this chapter is to give you some practical experience with JavaHelp before you study it in more detail. This chapter is "hands on": it walks you through the development process without lengthy explanations of the concepts behind it. The rest of this book then expands on the procedures you learn here, providing the detailed information you need to thoroughly understand JavaHelp development.

The HelpSet you'll build in this chapter—we'll call it *MyJavaHelp*—is even simpler than the Aviation sample introduced in Chapter 1, *Understanding JavaHelp*. But the procedures for building it are nearly the same as those for building a more complex HelpSet.

To best understand the structure of HelpSet data and navigation files, you should create them on your own. However, since the topic files are in basic HTML format, you might want to simply download the file set from this book's web page instead of creating them all from scratch: click on "Examples" at *http://www.oreilly.com/catalog/creatingjavahelp*.

At the end of this chapter, you will have a functioning JavaHelp system. To get you there, this chapter provides procedures to guide you through the following JavaHelp development processes:

- Creating the HelpSet's directory structure
- Creating HelpSet data and navigation files
- Creating help topic files
- Checking your work
- Testing the finished HelpSet

,ch02.29521 Page 19 Thursday, April 20, 2000 10:54 AM

Creating the HelpSet's Directory Structure

To access HelpSet files, JavaHelp depends on proper file and directory structure. Referring to a file by the wrong name or placing a file in a wrong directory, causes JavaHelp to fail when it tries to access the file. To ensure accurate file retrieval, start your project by setting up the HelpSet's entire directory structure. Figure 2-1 shows the directory and file structure you'll create for MyJavaHelp.

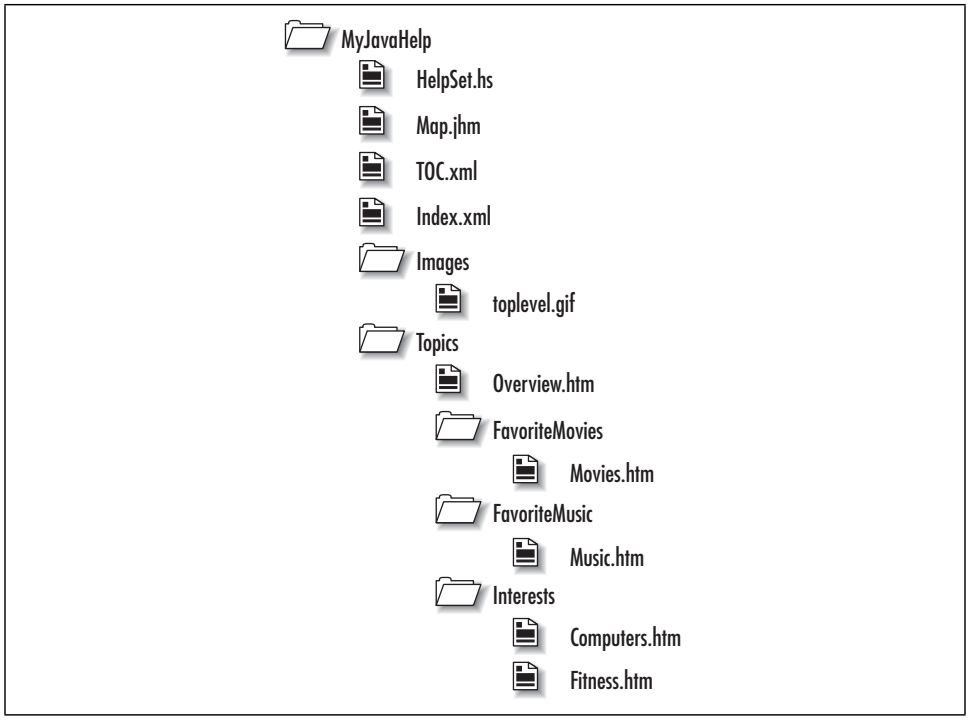

Figure 2-1. Structure of the MyJavaHelp HelpSet

In general, you can use whatever directory and file names you like. However, keep in mind the following:

- It's best to use the filename extensions shown in Figure 2-1.
- If you rename any of the data or navigation files (*Map.jhm*, *TOC.xml*, *TOC.xml*, *Index.xml*), you must edit the HelpSet file, *HelpSet.hs*, which contains references to these files.

Figure 2-1 shows a directory, *Topics*, which contains three subdirectories (*FavoriteMovies*, *FavoriteMusic*, and *Interests*). In general, you can use any number of directories and subdirectories, nested to any depth. Chapter 3, *Planning the Java-*

,ch02.29521 Page 20 Thursday, April 20, 2000 10:54 AM

Help Project, provides strategies for planning the organization of your file directories and presents options for arranging your files. In this chapter, you'll use a very simple structure.

Start the project by creating the directory structure shown in Figure 2-1. First, create *MyJavaHelp*, the master directory that will hold all the data for the HelpSet. Within the *MyJavaHelp* directory, create the following subdirectories:

- *Images* (to hold image files)
- *Topics* (to hold help topic files)

Place the *toplevel.gif* image inside the *Images* directory by copying it from the JavaHelp installation area (filename *jh1.1\Demos\hs\holidays\images\toplevel.gif*). This image file is the icon that appears at the top level of the TOC in the HelpSet Viewer.

In the *Topics* directory, create the subdirectories that will contain the actual help topic files: *FavoriteMovies*, *FavoriteMusic*, and *Interests*

NOTE Try to be consistent in devising directory and filenames. This will help you avoid spelling errors. The name of each subdirectory contains one or more capitalized words, with no intervening spaces. This style makes a directory name easy to read and clearly indicates the directory's contents

---

You now have the entire directory structure required for the MyJavaHelp HelpSet.

Creating HelpSet Data and Navigation Files

A HelpSet contains a set of topic files, along with a number of data and navigation files. The topic files are in HTML format. The data and navigation files are in XML (Extensible Markup Language) format. XML looks similar to HTML, but has application-specific tags, such as `<mapID>` and `<tocitem>`, instead of HTML's standardized document-formatting tags, such as `<head>` and `<p>`.

Each data and navigation file contains XML *elements*, most of which take the following form:

```
<element-name  attr-name="attr-value"  attr-name="attr-value"/>
```

Here, "attr" means "attribute." Each element can have any number of attributes, including none at all. The value of each attribute must be enclosed in single or double quotes. XML elements are like HTML *tags* (at least as far as JavaHelp is concerned).

,ch02.29521 Page 21 Thursday, April 20, 2000 10:54 AM

Creating the HelpSet File

Using your text editor, create the file *HelpSet.hs* in the *MyJavaHelp* directory, with the following contents. Be sure to get the capitalization correct: XML is case-sensitive!

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<!DOCTYPE helpset
  PUBLIC "-//Sun Microsystems Inc.//DTD JavaHelp HelpSet Version 1.0//EN"
         "http://java.sun.com/products/javahelp/helpset_1_0.dtd">

<helpset version="1.0">
  <title>My JavaHelp System</title>
  <maps>
    <mapref location="Map.jhm"/>
    <homeID>overview</homeID>
  </maps>
  <view>
    <name>TOC</name>
    <label>TOC</label>
    <type>javax.help.TOCView</type>
    <data>TOC.xml</data>
  </view>
  <view>
    <name>Index</name>
    <label>Index</label>
    <type>javax.help.IndexView</type>
    <data>Index.xml</data>
  </view>
</helpset>
```

This file specifies the names of the other data and navigation files for the MyJavaHelp HelpSet: *Map.jhm*, *TOC.xml*, and *Index.xml*. You'll create these files in the sections that follow.

NOTE In this file, as in all XML-format configuration files, it doesn't matter whether you indent elements with spaces or tabs. But it *does* help humans to read the file, since it shows how some elements are logically contained within others.

Creating the Map File

The map file assigns a unique keyword (called a *map ID*) to each topic file in the HelpSet, using `mapID` elements:

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

,ch02.29521 Page 22 Thursday, April 20, 2000 10:54 AM

Creating a map file can be long and tedious for a HelpSet with many topic files, but it is easy for MyJavaHelp, which is very small. For simplicity, use each topic filename (without the extension) as its map ID.

Using a text editor, create the file *Map.jhm* in the *MyJavaHelp* directory, with the following contents:

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<!DOCTYPE map
  PUBLIC "-//Sun Microsystems Inc.//DTD JavaHelp Map Version 1.0//EN"
         "http://java.sun.com/products/javahelp/map_1_0.dtd">

<map version="1.0">

  <mapID target="toplevelfolder" url="Images/toplevel.gif" />

  <mapID target="overview" url="Topics/Overview.htm"/>

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

  <mapID target="movies" url="Topics/FavoriteMovies/Movies.htm"/>

  <mapID target="music" url="Topics/FavoriteMusic/Music.htm"/>

</map>
```

The URLs in this map file are *relative* pathnames—they are relative to the directory in which the map file resides. Thus, these URLs rely on the *Topics* subdirectory being in the same directory as the *Map.jhm* file (the *MyJavaHelp* directory). However, if an image or topic file were in another directory or at some other site on the Web, you would have to use an *absolute* path, as shown in the following example:

```
<mapID target="overview" url=" http://www.kevinlewis.com/download/overview.htm"/>
```

NOTE In a map file, you must use forward slashes (`/`) instead of backslashes (`\`)in the values of `url` attributes. That is, you specify a URL just as you would in a web browser. Also, be aware of case-sensitivity. Windows systems are case-insensitive in reading filenames (and thus, URLs); Unix systems, however, are case-sensitive. It's best to be conservative: always specify file and directory names paying close attention to case.

,ch02.29521 Page 23 Thursday, April 20, 2000 10:54 AM

Creating the Navigation Files

Navigation is an important part of any online help system. The MyJavaHelp HelpSet you are creating in this chapter is quite small, so users could easily find each topic even if you provided only a table of contents (TOC). But imagine trying to find a topic in a HelpSet that contains 200, 500, or more topics. With larger HelpSets, the user is dependent on a complete and usable navigation system.

Even though your HelpSet is small, you should still create a complete navigation facility, including at least a TOC and index. You could also create a word-search index, but I'll defer that discussion until Chapter 5, *Creating HelpSet Data and Navigation Files.*

Creating the Table of Contents file

The TOC file uses a different set of XML elements from the map file. Using your text editor, create the file *TOC.xml* in the *MyJavaHelp* directory, with the following contents:

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<!DOCTYPE toc
  PUBLIC "-//Sun Microsystems Inc.//DTD JavaHelp TOC Version 1.0//EN"
         "http://java.sun.com/products/javahelp/toc_1_0.dtd">

<toc version="1.0">
<tocitem image="toplevelfolder" target="overview" text="My JavaHelp System">
    <tocitem text="Interests">
      <tocitem target="fitness" text="Fitness"/>
      <tocitem target="computers" text="Computers"/>
    </tocitem>
    <tocitem text="Favorite Movies">
      <tocitem target="movies" text="Favorite Movies"/>
    </tocitem>
    <tocitem text="Favorite Music">
      <tocitem target="music" text="Favorite Music"/>
    </tocitem>
</tocitem>
</toc>
```

(This is an example of a place where indentation style is critical to human readability. "When nesting gets deep, the tough reach for the Tab key.") You probably recognize the map IDs you created in the preceding section. I'll go into more detail about the TOC and other navigation files in Chapter 5. For now, note that:

- The `text` attribute of a `tocitem` element specifies the title to appear in the TOC as displayed by the HelpSet Viewer.

,ch02.29521 Page 24 Thursday, April 20, 2000 10:54 AM

- Some `tocitem` elements are nested within other `tocitem` elements. Nesting of elements represents the hierarchy of the TOC.

Creating the Index File

The index file resembles the TOC in structure and format. But keep in mind that JavaHelp doesn't automatically alphabetize the index items. The order of items in the index file is the order in which the items will appear in the HelpSet Viewer. You'll want to allocate time (and/or help-authoring tools, as described in Chapter 9, *Using Third-Party Help-Authoring Tools*) for ensuring that the index items are in alphabetical order in the index file.

Using your text editor, create the file *Index.xml* in the *MyJavaHelp* directory, with the following contents:

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<!DOCTYPE index
  PUBLIC "-//Sun Microsystems Inc.//DTD JavaHelp Index Version 1.0//EN"
         "http://java.sun.com/products/javahelp/index_1_0.dtd">

<index version="1.0">
  <indexitem target="computers" text="computer interests"/>
  <indexitem text="favorites">
    <indexitem target="movies" text="movies"/>
    <indexitem target="music" text="music"/>
  </indexitem>
  <indexitem target="fitness" text="fitness interests"/>
  <indexitem text="interests">
    <indexitem target="computers" text="computers"/>
    <indexitem target="fitness" text="fitness"/>
  </indexitem>
  <indexitem target="movies" text="movies, favorite"/>
  <indexitem target="music" text="music, favorite"/>
  <indexitem target="overview" text="overview"/>
</index>
```

As in the TOC file, the `text` attribute specifies the item to be displayed in the HelpSet Viewer. Nesting of `indexitem` elements defines primary and secondary index items. In this file, `interests` is a primary index item, and `computers` and `fitness` are its secondary items.

Creating Help Topic Files

In JavaHelp, the contents of each help topic is defined by a file in HTML format. In this sense, JavaHelp's HelpSet Viewer is very much like a web browser.

,ch02.29521 Page 25 Thursday, April 20, 2000 10:54 AM

In this section, you'll create the individual help topic files. As I mentioned before, if you want to save yourself the time and effort of typing a lot of HTML-format text, simply use the HTML topic files available under "Examples" on this book's web page.

You've already declared, in the map file, both the names and the directory locations of the following MyJavaHelp topic files:

- File *Overview.htm* in the *Topics* directory (see Example 2-1)
- File *Fitness.htm* in the *Interests* subdirectory of the *Topics* directory (see Example 2-2)
- File *Computers.htm* in the *Interests* subdirectory of the *Topics* directory (see Example 2-3)
- File *Movies.htm* in the *FavoriteMovies* subdirectory of the *Topics* directory (see Example 2-4)
- File *Music.htm* in the *FavoriteMovies* subdirectory of the *Topics* directory (see Example 2-5)

Example 2-1. Overview.htm

```
<html>
<head>
<title>Overview</title>
</head>
<body>
<h1>Overview</h1>
<p>Welcome to my JavaHelp system. In this help system I discuss
my interests in the following topics:
<ul>
  <li>Fitness</li>
  <li>Computers</li>
  <li>Movies</li>
  <li>Music</li>
</ul>
</body>
</html>
```

Example 2-2. Fitness.htm

```
<html>
<head>
<title>Fitness</title>
</head>
<body>
<h1>Fitness</h1>
<p>I enjoy the following fitness activities:
<ul>
```

,ch02.29521 Page 26 Thursday, April 20, 2000 10:54 AM

Example 2-2. Fitness.htm (continued)

```
  <li>Running</li>
  <li>Biking</li>
  <li>Hiking</li>
  <li>Swimming</li>
</ul>
</body>
</html>
```

Example 2-3. Computers.htm

```
<html>
<head>
<title>Computers</title>
</head>
<body>
<h1>Computers</h1>
<p>I use computers every day for both work and pleasure.</p>
</body>
</html>
```

Example 2-4. Movies.htm

```
<html>
<head>
<title>Favorite Movies</title>
</head>
<body>
<h1>Favorite Movies</h1>
<p>The following movies are some of my favorites:
<ul>
  <li>Titanic</li>
  <li>Sphere</li>
  <li>Halloween</li>
  <li>48 Hours</li>
</ul>
</body>
</html>
```

Example 2-5. Music.htm

```
<html>
<head>
<title>Favorite Music</title>
</head>
<body>
<h1>Favorite Music</h1>
<p>I enjoy the following types of music:
<ul>
  <li>Rock and roll</li>
```

,ch02.29521 Page 27 Thursday, April 20, 2000 10:54 AM

Example 2-5. Music.htm (continued)

```
  <li>Pop</li>
  <li>Easy listening</li>
  <li>Jazz</li>
</ul>
<p>You should also see my list of <a href="../FavoriteMovies/Movies.htm">
favorite movies</a>.</p>
</body>
</html>
```

Checking Your Work

That's all there is to creating the MyJavaHelp HelpSet. Because the directory and file structure is crucial for successfully running the help system, you should double-check your HelpSet's structure. Make sure it matches the structure shown in Figure 2-1.

Testing the Finished HelpSet

You can view your HelpSet the same way you viewed the Aviation HelpSet in the preceding chapter. Start the *hsviewer* utility that comes with JavaHelp and specify *HelpSet.hs* in the *MyJavaHelp* directory as the HelpSet file.

Try out the different navigation components. If the HelpSet Viewer is not already displaying the TOC, click the **TOC** tab to view it. Notice the top-level directory image on the first line in the TOC. This image is the *toplevel.gif* file you copied to your *Images* directory. Open a folder in the TOC, and click one of the topics to display it in the content pane.

Now click the **Index** tab. In the **Find** box, type the word `interests` and then press the Enter or Return key. Notice that JavaHelp highlights the entry in the index. Click any index entry to display its corresponding topic in the content pane.

Congratulations! You (and JavaHelp) have just created a functional help system.