

JavaHelp Overview

Sun Microsystems' JavaHelp is a new help technology for developing multiplatform online help. The promise of running JavaHelp on any platform is exciting, but implementing a basic JavaHelp system can be difficult. This document provides a real-world overview of JavaHelp, its components, and how to begin building JavaHelp systems.



Scriptorium Publishing Services, Inc.
P.O. Box 12761, Research Triangle Park, NC 27709
919-481-2701 or sales@scriptorium.com
<http://www.scriptorium.com>

Getting Started

JavaHelp runs inside the Java Virtual Machine (JVM). The JVM is different from the Java that runs inside your web browser, so you'll need to download some software from Sun to get started with JavaHelp. You'll find the general Java site at:

<http://java.sun.com>

JavaHelp information is at:

<http://java.sun.com/products/javahelp/index.html>

Unfortunately, the software gets updated so frequently that it's impossible to predict exactly which file you should download. However, you'll need two basic components:

- Java Runtime Environment (JRE)
- JavaHelp

The JRE gives you all the components for running Java applications on your system. For some platforms, Sun may have a combination JRE/JavaHelp download file, so look for that download first. Otherwise, you'll need to download them separately.

If you want to view JavaHelp in your browser instead of embedded in an application or displayed in a separate JavaHelp window, you need to install the Java Plug-In. Sun provides the Java Plug-In so that users can update their browsers to read the most current version of Java. JRE 1.2.2 includes the Java Plug-In, but you can also download it separately from <http://java.sun.com/products/plugin/>.

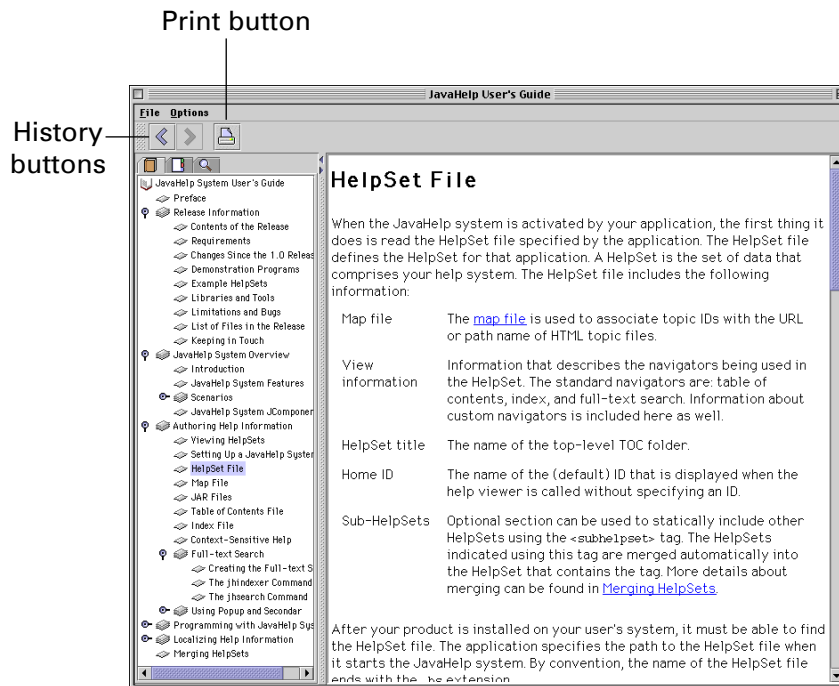
If you plan to develop Java applications, download the Java Developers Kit (JDK); it also includes the JRE, so you don't have to download it separately.

In summary, you need to download the JRE, JavaHelp, and the Java Plug-In (if you want to view JavaHelp in your browser) and install the software. Installation is quite straightforward, and it's probably a good idea to reboot after you finish installing the software.

Checking JavaHelp installation

After you install the JRE/JDK and JavaHelp components, you'll probably want to make sure that it's working! Look in the JavaHelp folder for the *JavaHelp User's Guide* (JHUG, pronounced "jay-hug"). Double-click the JHUG to start it. If you're on a PC, you can select the Start menu, JavaHelp, then JavaHelp User's Guide. If all is well

with your Java environment, you should see the JavaHelp User's Guide in a JavaHelp window.



The JavaHelp window, like many other help technologies, displays the navigation information (table of contents, index, or search) in the left frame and the content on the right. At the top of the window, a small toolbar provides history and printing buttons.

History buttons

The history buttons work just like the Forward and Back buttons in a web browser. They allow you to move back and forward through topics that you have viewed.

The history buttons are different from browse sequences in WinHelp, which move you through a predetermined path. The result of clicking one of the history buttons depends on which topics you just viewed.

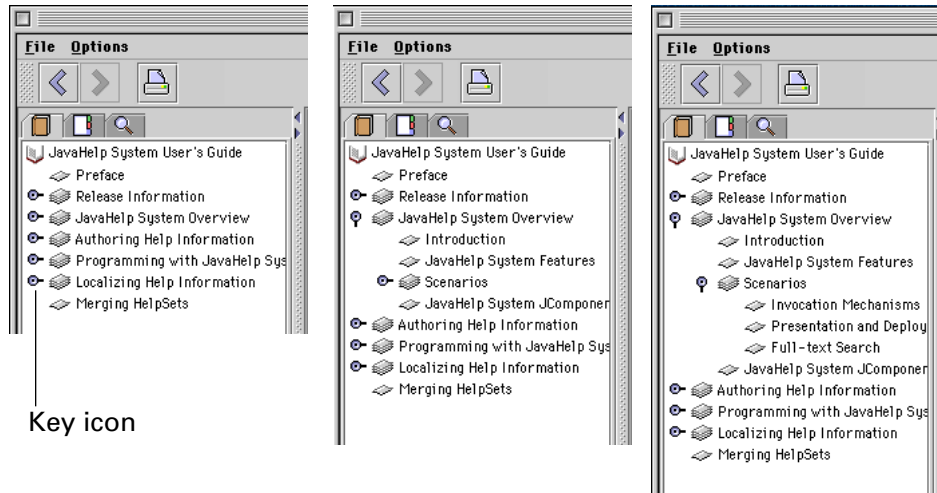
Printing

You can print the page in the right frame by clicking the Print button. This opens the Print dialog box. In Windows, there's also a Print Setup button that lets you set the margin and page sizes.

Users have reported particularly long print times, which is a JDK 1.2 bug. JDK 1.3, which should be released in the first quarter of 2000, will fix this issue.

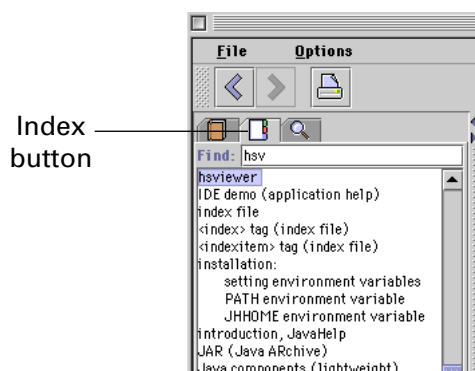
Table of contents

The table of contents is the first page displayed in the left frame. You can click the key icons to show and hide parts of the table of contents.



Index

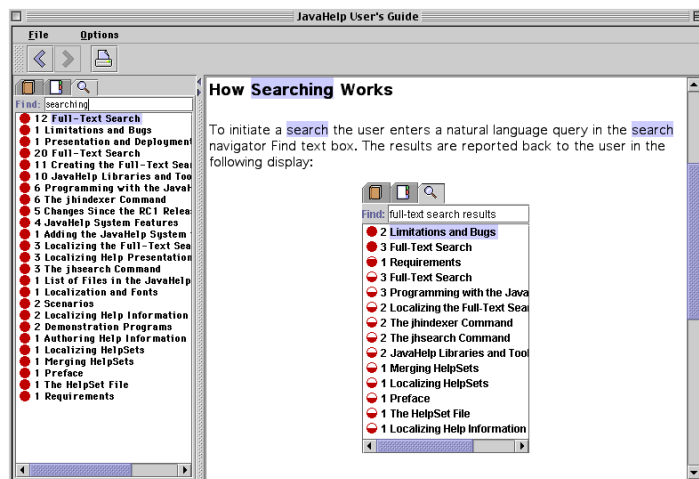
JavaHelp creates an index similar to other help systems. Clicking the tab with the white book icon displays the index. Single- and multilevel entries are displayed. For example, if you click the first-level entry “installation” in the following figure, nothing happens. You must click a second-level entry (“setting environment variables,” for example) for the information to be displayed in the right frame. You can also type a keyword in the Find window, press Enter, and see the first instance of that keyword highlighted in the index. The content is displayed in the right frame. Press Enter again to see if the keyword is found elsewhere. If your computer beeps, there are no more matches.



Search

The JavaHelp full-text search differs from searches in other help formats by its weighted results. Click the magnifying glass tab to display the search page, type the search term in the Find window, and press Enter. The results are displayed in the left frame and are ranked in two ways. The number next to the result indicates how relevant the result is to your search. A high number means that the entry is more relevant than entries with low numbers. The red circle also indicates importance—the more complete the circle, the higher the relevance.

Each instance of the search term and its stem words are highlighted in the right frame. For example, if you search for the word “searching,” “search” is highlighted in the text along with the complete search term.



File menu

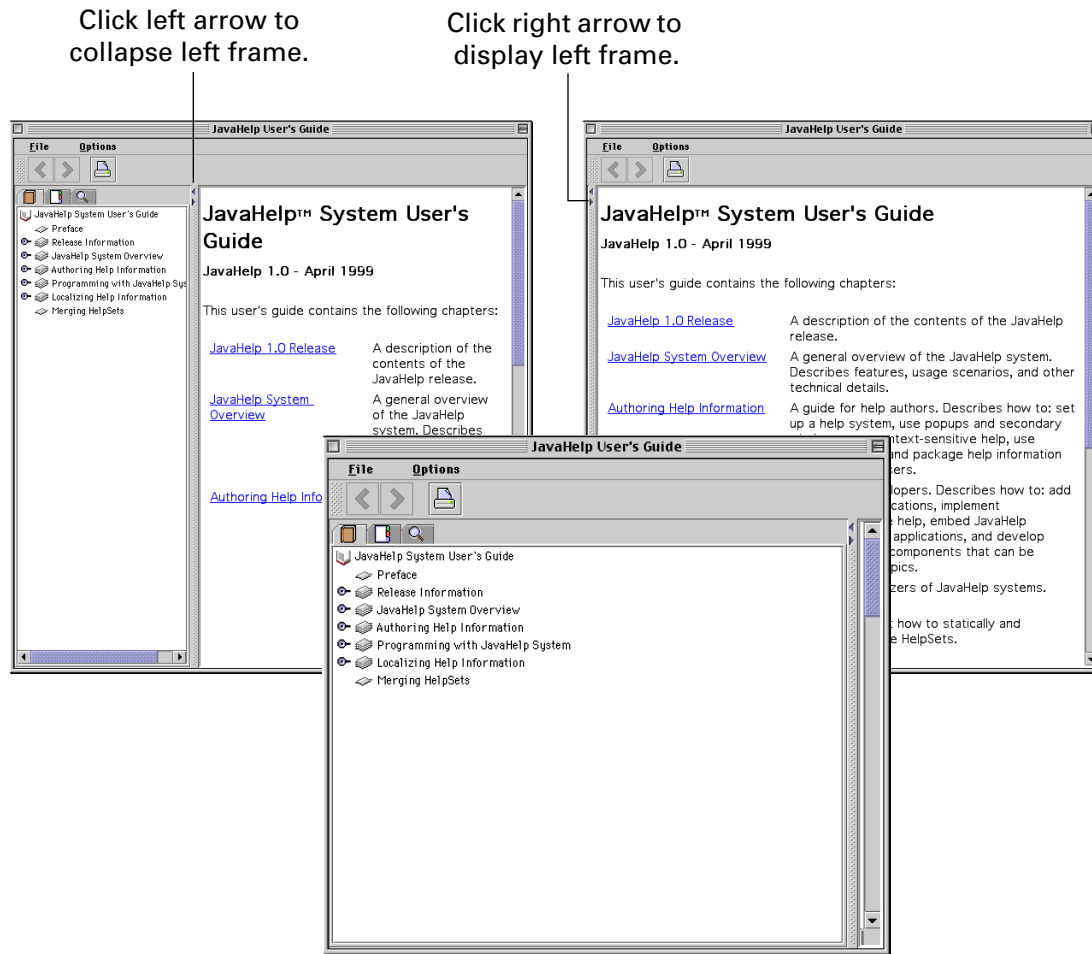
The file menu provides two options—opening a file or web page and exiting the program. Select the File menu, then Open Page, and the Open Page window is displayed. You can click Choose File to open a local file, or you can type a URL in the URL window to open a web page in the JavaHelp application window. For example, you may want to open a web page in the JavaHelp window instead of opening your browser.

Options menu

In the Options menu, you can select Set Font to change the font used in the help file. Currently, this feature doesn't work as expected—only the font in the left frame changes. JDK 1.3 will fix this problem.

Expanding and collapsing the left frame

Between the frames, you'll find two arrows that let you expand or collapse both frames. Clicking the left arrow collapses the left frame and displays a full view of the content. Clicking the right arrow returns the frame to its original position. If you want a full view of the left frame, click the right arrow first, then click the left arrow to return the frame to its original position.

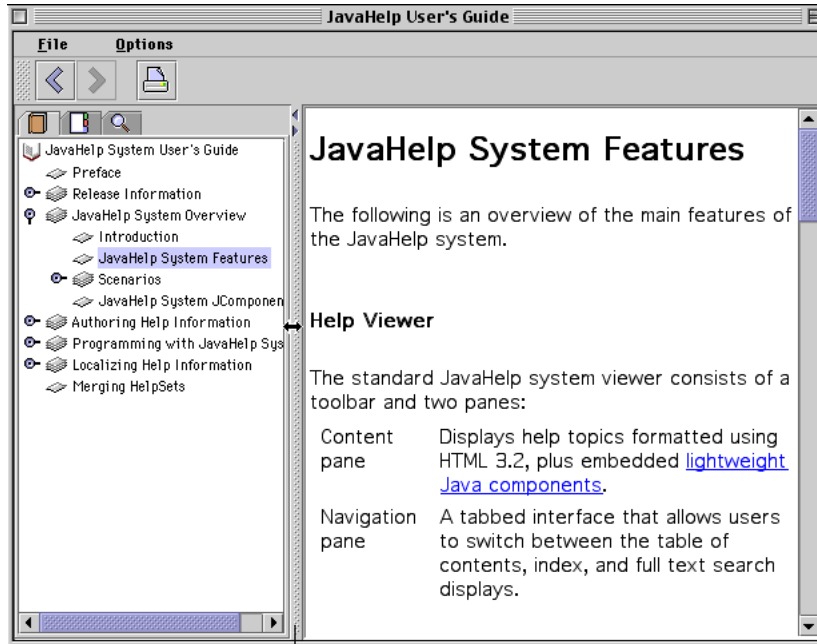


View only the contents by clicking the right arrow then the left arrow.

Resizing the frames and window

Both the JavaHelp window and the frames are resizable. When you open JavaHelp, you may need to drag the edge of the window to resize it and make the content easier to read. To resize the frames, you drag the separator to the left or right. These changes are not saved when you exit the application.

If you develop help in JavaHelp 1.0, you can't control the initial size of the window. Instead, you must resize the window each time you open the JavaHelp file.



Drag the separator to
resize the frames.

Understanding the JavaHelp component files

The information presented in JavaHelp files is, for the most part, regular HTML files and GIF files. JavaHelp also uses several additional files:

- HelpSet file (.hs)
- Mapping file (.jhm)
- Table of contents file (.xml)
- Index file (.xml)
- Content files (.html and .gif)

HelpSet file

The HelpSet file is the "container" file for the JavaHelp project. It tells JavaHelp where the help files reside, what contents and index files to use, and gives additional information.

The HelpSet file is an XML file. Here is a typical HelpSet file:

```
<?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 -->
<title>JavaHelp User's Guide</title>

<!-- maps -->
  <maps>
    <homeID>top</homeID>
    <mapref location="jhug.jhm" />
  </maps>

  <!-- views -->
  <view>

    <name>TOC</name>
    <label>Table Of Contents</label>
    <type>javax.help.TOCView</type>
    <data>jhugTOC.xml</data>
  </view>

  <view>
    <name>Index</name>
    <label>Index</label>
    <type>javax.help.IndexView</type>
    <data>jhugIndex.xml</data>
  </view>
</helpset>
```

The first line identifies this file as an XML file. The <!DOCTYPE> element specifies the location of the DTD file.

This title is displayed in the JavaHelp title bar.

Maps are for context-sensitive help. The <homeID> element specifies which topic is displayed when you first open the file by referencing the mapping file, `jhug.jhm`. The <view> elements let you define your table of contents, index, and other views.

Mapping file

The mapping file lists the targets and URLs for elements such as the button graphic used in the left frame, pages that display in the right frame, index IDs, and context-sensitive help IDs. The mapping file is an XML file. The typical mapping file can be quite long, so here's an excerpt:

```
<?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="chapter" url="images/chapTopic.gif" />
<mapID target="topic" url="images/topic.gif" />
```

In this example, `toplevel.gif` is the graphic used for each first-level TOC entry.

```
<mapID target="openbook" url="images/openbook.gif" />
<mapID target="top" url="jhug.html" />
<mapID target="rel.release" url="release/release.html" />
<mapID target="rel.install" url="release/install.html" />
</map>
```

Jhug.html is the first page to display in the right frame. Remember that the HelpSet file created a reference to the "top" target.

Table of contents file

The table of contents file (TOC) is an XML file that creates the TOC window. Two main tags are used in the TOC: `<toc>`, which defines the TOC, and `<tocitem>`, which defines each entry. To create different levels in the TOC, you insert indents before the `<tocitem>`. In the example below, Release Information is a first-level entry. The indented `<tocitem>` below it, Contents of the Release, is a second-level entry.

Each `<tocitem>` contains one or more words that define the entry further. Text indicates the text for the TOC entry. Image indicates the image displayed to the left of the entry. And target indicates the page, defined in the mapping file, that is displayed in the right frame when the user clicks the text. For example, when a user clicks Release Information, the target `rel.release` is called. This target is defined in the mapping file as `release/release.html`, so `release.html` is displayed in the right frame.

The following is an excerpt from the TOC file.

```
<?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 text="JavaHelp User's Guide" image="openbook">

<tocitem text="Preface" image="topic" target="top"/>

<tocitem text="Release Information" image="chapter"
target="rel.release">
  <tocitem text="Contents of the Release" image="topic"
target="rel.contents"/>
  <tocitem text="Requirements" image="topic"
target="rel.requirements"/>
  <tocitem text="Changes Since the 1.0 Release" image="topic"
target="rel.changes"/>
  <tocitem text="Limitations and Bugs" image="topic"
target="rel.limitations"/>
  <tocitem text="List of Files in the Release" image="topic"
target="rel.file_list"/>
</tocitem>
</toc>
```

The first `tocitem` defines the first line of text and an image, which is specified in the mapping file. The third `tocitem` defines the top-level entry and an image. Indented lines define items in the TOC.

Index file

The index file, another XML file, contains index entries and their targets. Multilevel index entries are created using indentation levels, the same as in the TOC file.

In the following example, the first-level entry, adding an existing project, is followed by two second-level entries—naming the project and naming the storage directory. The targets for these entries are defined in the mapping file. They specify the file displayed in the right frame after the user clicks an index entry. For example, when the user clicks naming the project, the target `proj.importdirectory` is called. This target is defined in the mapping file as `docs/import.html`, so `import.html` is displayed in the right frame.

Here's an excerpt from the index file:

```
<?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">

<!-- A -->
  <indexitem text="adding an existing project">
    <indexitem text="naming the project"
target="proj.importdirectory"/>
    <indexitem text="naming the storage directory"
target="proj.importdirectory"/>
  </indexitem>

<!-- B -->
  <indexitem text="background color, source editor"
target="edit.syntax"/>

  <indexitem text="BeanInfo file" target="proj.beansproject"/>

</index>
```

Creates a first-level index entry for "adding an existing project," then specifies second-level entries and their targets.

Content files

Content files can be HTML or GIF files. JPEG files may display as distorted images, so stick with GIFs.

Implementing full-text search (jhindexer)

JavaHelp provides a full-text search feature. To implement full-text search for your project, you first create all of the JavaHelp files, then you run the `jhindexer` utility. You'll find this executable in the `javahelp/bin` directory of the JavaHelp system installation.

The command line syntax looks like this:

```
jhindexer [options] [file | folder]*
```

Using the options attributes, you can specify the following:

- You can configure a list of stopwords to create more useful search results. Stopwords are words that are irrelevant in searches, such as “the,” “and,” “or,” and “was.” When you provide your own list of stopwords, the default stopwords list isn’t used.
- You can name the configuration file. This is the file in which you change the path name of the files being indexed, specify the files you want indexed, and provide your list of stopwords.
- You can specify the language used in your project.
- You can specify that the jhindexer messages be written to file.
- You can indicate a folder for the output (default is JavaHelpSearch).
- You can specify that verbose messages, such as a list of files included in the full-text search, be displayed while the database is being processed. This attribute is also useful for debugging.

For more information on each attribute, see the *JavaHelp User’s Guide*.

Implementing context-sensitive help

In JavaHelp, you can have four kinds of context-sensitive help:

- Window level, accessed by pressing F1
- Field level, accessed by clicking an object
- A Help menu, accessed by selecting the Help menu
- A Help button, accessed by clicking a Help button on the toolbar

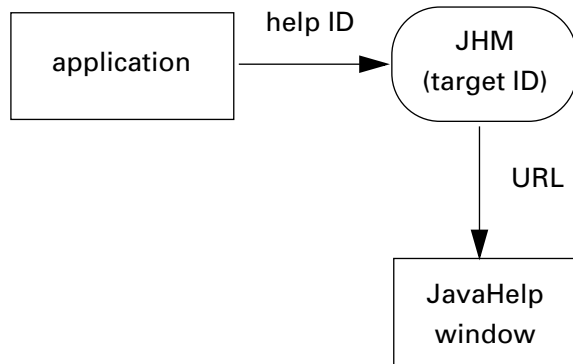
The JavaHelp installation includes the HelpBroker. This interface helps you write code and hides much of the complex operations.

You can also write the code yourself, which isn’t as daunting as it sounds. Each interface window you’ve documented has a target ID. When the user accesses help (using one of the preceding methods), this target ID is called, and the associated help file is displayed in a separate window. You can assign the target ID, or the developer can assign it.

In the Mapping file, you include each target ID and the URL of its help file. Your code may look something like this:

```
<mapID target="4554" url="docs/walk_dog.html" />
```

In this example, the target ID 4554 is mapped to the help file `walk_dog.html`. When the user reads about walking dogs in your application and accesses online help for that topic, `walk_dog.html` will be displayed in the JavaHelp window. Here's the basic process:



Delivering JavaHelp

After you create your JavaHelp, you have the option of compressing it into a *Java ARchive* (JAR). A JAR file consists of a zipped file and sometimes a Manifest file (named `META-INF/MANIFEST.MF`) that describes the contents of the JAR file.

The JAR command is located in your JDK/bin folder. The command line syntax looks like this:

```
jar [ctxvfm] [jar-file] [manifest-file] files ...
```

The attributes `[ctxvfm]` include specifying the JAR file name, creating a JAR table of contents, generating verbose error output, and other basic functions.

There are a few reasons that you may not want to compress your help files into a JAR file. Developers have reported that compressing the files causes the full-text search not to work, a problem that JDK 1.3 will fix. In addition, if you leave the HelpSet file out of the JAR file, you can more easily update the user's help files. You simply have to ship them a new HelpSet file and new JAR files (containing your new content) instead of shipping the entire installation.

JavaHelp authoring tools

Several JavaHelp authoring tools are on the market. The tool you choose depends on several factors, one of which is the format of the source material. For example, FrameMaker users may want to use Quadralay WebWorks Publisher because its specialty is FrameMaker conversion. The Sun JavaHelp web site lists several tools. Here are some of the most popular tools for technical writers along with URLs so you can find more information and pick the tool best suited for your situation:

- RoboHelp Office (<http://www.bluesky.com>)
- Quadralay WebWorks Publisher (<http://www.quadralay.com>)
- ForeFront ForeHTML (<http://www.ff.com>)
- WexTech Systems Doc-to-Help (<http://www.wextech.com>)

Contacting us

Scriptorium Publishing Services, Inc. excels at transforming complex technical ideas into clear, concise documents. Our clients, who range from start-ups to Fortune 500 companies, rely on us for the full spectrum of technical publishing services—everything from turnkey documentation to specialized technical editing and consulting. Our expert, talented staff thrives on working with challenging new technology in a deadline-driven environment.

We are WebWorks Publisher trainers and offer training and consulting services for FrameMaker and WebWorks Publisher.

If you have any questions about Scriptorium Publishing Services, Inc., contact:

Scriptorium Publishing Services, Inc.
P.O. Box 12761
Research Triangle Park, NC 27709
919-481-2701
sales@scriptorium.com
<http://www.scriptorium.com>