JavaHelp™ White Paper

A Platform-Independent Help System

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303
1 (800) 786.7638
1.512.434.1511

Please
Recycle

Contents

CHAPTER 1

Introduction

Today, Java™ applets and applications are proliferating rapidly in every imaginable computing environment. For users to be successful with these new applications, it is important to get up and running quickly. To this end, an online help facility can dramatically reduce the learning cycle for most applications. Application help assists users stay productive while keeping pace with the constant flow of new features. However, until recently, no standard help system was well integrated into the Java development platform.

Now, Sun Microsystems delivers the JavaHelp™ application programming interface (API) and help system reference implementation for creating cross-platform, extensible application help. JavaHelp software-based systems offer a powerful solution for information architects, help and documentation authors, and application developers by providing unprecedented flexibility in a full-featured, open standard that is easy to use. JavaHelp systems leverage the many benefits of the Java language, so they can be used to present online help for any software application deployed on any platform — not just for those applications written in the Java language. In addition, a JavaHelp system is highly effective when used by itself — without association to a software application — as a delivery mechanism for Web-centric documentation in categories such as human resources, technical support, and other reference materials.

The JavaHelp reference implementation is 100% Pure Java™ and is an extension to the JDK™ 1.1.x, Standard Edition and the Java 2 Platform, Standard Edition. Based on Java Foundation Classes (JFC), the JavaHelp API offers a easy-to-use interface that enables developers and help system authors to quickly add online help to applications, Internet sites, and corporate intranets.

CHAPTER 2

Market Overview of Help Systems

Application Help

With the staggering number of new Java software-based applications, enterprise managers and software vendors recognized that application help is a critical element in reducing user learning curve and moving users into productive activity. When properly implemented, an application's help facility is intuitive and allows users to navigate, search, and quickly display help information.

Limited Help Options for Java Software-Based Applications

Until now, developers of Java software-based applets and applications had few attractive options for implementing a help system, including:

- No help system – The simplest approach is to let users figure out how everything works. The application developer ignores the consequences.
- Use standard HTML – An expedient, low-cost option that has limited capability and is not user friendly; provides basic text information in spartan modes of operation, but is not intuitive or easy to navigate.
- Custom system – Requires a substantial investment of resources which can burden the application's development process. Investment in custom help systems is often cut short by other pressing demands such as time to market or budget constraints.
- Third-party solution – Although likely to be faster and more capable than other solutions, it is also more likely to be proprietary. Long-term support of proprietary technology is also a consideration with third-party solutions.

None of these alternatives offers the benefits provided by systems built using the JavaHelp API. JavaHelp software-based systems offer rich functionality, and are designed based on open, industry standards. The JavaHelp API elegantly addresses all of the major challenges facing help authors and application developers today.

Cross Platform Help — Moving from WinHelp to JavaHelp

When considering a move from Microsoft WinHelp to an HTML-based help system, JavaHelp software is the clear choice. Using a help system that is not based on Java technology defeats the cross-platform advantage of HTML content because it can lock companies into a single platform. In addition, JavaHelp software already offers many features found in WinHelp, and extends this feature set further. JavaHelp software-based systems preserve the benefits of cross-platform content and the help system itself.

On-Line Documentation

Since Web-based documentation and help systems have similar requirements for display, navigation, and search capability, the JavaHelp system is ideal. Companies find Web-based distribution cost-effective and easier to manage. In fact, many companies plan to use the JavaHelp API to establish a uniform interface for displaying and navigating through a vast array of internal and external documents. Now that distributing information via the Web has become the de facto standard for corporate intranets and the Internet, the JavaHelp system can be used effectively for delivering information in categories such as online employee policy manuals, training, reference materials, and technical support. JavaHelp software is the ideal solution for building an online documentation system because it is fully featured, based on open standards, and runs on a wide variety of systems and platforms.

CHAPTER 3

Benefits of JavaHelp

An Array of Benefits for Help System Authoring

An online help system completes an application by providing users with the information they need to use the program effectively. Due to the inherent power of the JavaHelp API, developers enjoy a wide range of options when creating help and documentation systems. As with other Java applets and applications, JavaHelp systems enjoy the many benefits associated with the Java platform.

Cross-Platform, Cross-Browser Solution

JavaHelp systems run inside any Java software-enabled browser such as Netscape Navigator™ or Microsoft Internet Explorer, for easy access by virtually anyone on any platform including the Solaris™ Operating Environment, Windows 95, Windows 98, Windows NT, and Mac OS, among others. JavaHelp systems also run as standalone applications right on the desktop of any computer running the Java virtual machine, which is provided free as part of the JDK software.

Because the JavaHelp API can be used to develop a help system for applications written in the Java language or other languages, developers and authors can write a single common help system that can be deployed for applications written in various programming languages. Authors can write the help system once using the JavaHelp API and integrate it into C++ or SmallTalk applications.

Extensible

Since all graphical elements are presented as *swing* components based on Java Foundation Classes (JFC), JavaHelp components — viewer, navigators, search engine and toolbar — are easily combined or replaced by third-party offerings to create a help system with the desired characteristics. Developers can also include components that incorporate voice or sound into the viewer. This extensibility gives developers and help authors great flexibility and control in designing help systems to meet specific application requirements. With component-based architectures, help authors can also benefit from reuse and simplified RAD constructions.

Network Friendly

JavaHelp systems are optimized for use on the Internet and many other networked environments. Information designers are rapidly recognizing the important benefits of networked help systems. With JavaHelp software-based systems, help data, the viewer, navigators, and search components can be stored locally at the client, on a network server, or distributed between the two. This provides developers and network administrators with the flexibility to design systems that can be tuned for optimal performance based on different deployment scenarios.

Strong Authoring Tool Support

As an open, industry standard, the JavaHelp API is strongly supported by major help authoring tool vendors, including Blue Sky, Forefront, Quadralay, Solutionsoft, and WexTech. JavaHelp is a complete solution that eliminates the need for non-standard or limited-feature help systems.

Easy Localization

When applications are localized to support users in different countries, help systems must also be localized. Because they are written completely in the Java language, which supports internationalization (I18N), JavaHelp systems can be quickly localized for deployment into international markets.

CHAPTER 4

JavaHelp Features

The JavaHelp API provides a reference implementation of a comprehensive help system, enabling developers and authors to quickly and easily add online help to such as applications. Components such as navigational controls (table of contents, indices, full text-search capabilities), viewers, and toolbars enable authors to tailor a help system to meet specific application requirements.

JavaHelp Help Viewer Navigational Controls

JavaHelp software uses leading-edge XML technology for navigational control data, including table of contents, index, helpset definition, and map files. Rapidly gaining in popularity, XML is a Web technology that simplifies the process of data structuring.

The standard JavaHelp viewer consists of a toolbar and two panes:

- Content pane – Displays help topics formatted using HTML 3.2 plus frames, tables, and lightweight components.
- Navigation pane – A tabbed interface that allows users to switch between table of contents, index, and full text-search displays. Navigators are synchronized with the content viewer, so the user is always aware of their relative locations and options.

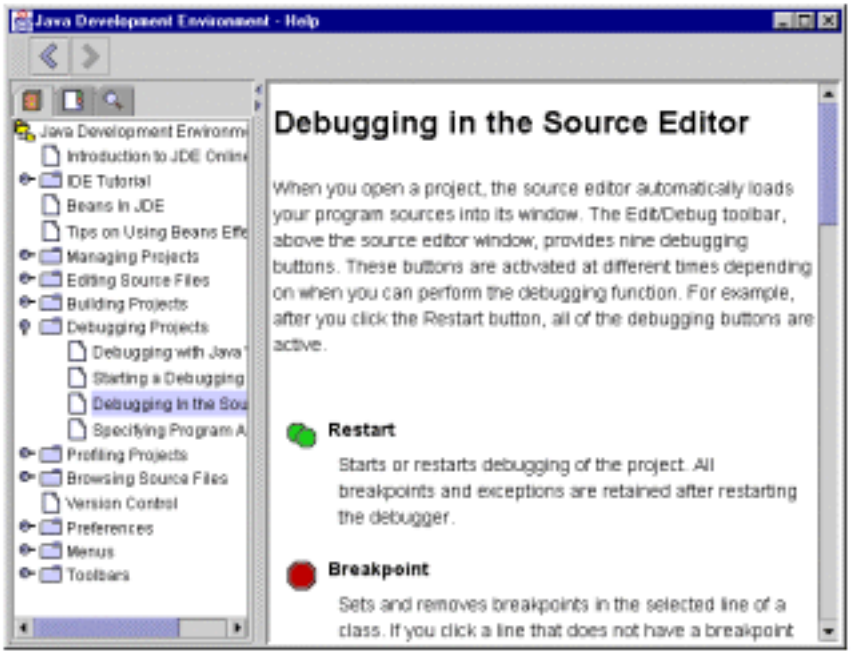

FIGURE 1 JavaHelp Help Viewer: JavaHelp software provides a navigation pane on the left and a content pane on the right, and provides access to navigators, including table of contents, indices, and full text-search capabilities.

Table of Contents

The JavaHelp table of contents is a collapsible and expandable display of topics in the help system, and is synchronized to scroll with the information displayed in the content pane. It supports unlimited levels and merging of multiple tables of contents. The underlying file format adheres to World Wide Web Consortium (W3C) standards.

Index

The index navigator provides a convenient lookup area for keywords or functions within the parent application or document, and supports merging of multiple indices. Its underlying file format adheres to W3C standards.

Full Text Search

JavaHelp software features a fast, cutting-edge search engine that can be used to search for help data on client-based, server-based, or distributed help systems in many different network environments. Users can ask for multiple-word queries; matches are ranked for relevancy using relaxation rules.

Encapsulation and Compression

To reduce the size of help files, JavaHelp files may be encapsulated and compressed into the Java Archive (JAR) format. Developers and authors can decide what information is best encapsulated into a helpset, enabling users to pull in the appropriate small or large helpsets based on the anticipated usage in specific sections of an application. The JavaHelp system also works with information not compressed into JAR files. This option allows authors to view files during development without requiring encapsulation or compression.

Merging Different Helpsets

Because today's applications often utilize several different interacting components, it can be helpful to combine these helpsets into an integrated, streamlined presentation. With the JavaHelp system, overlapping helpsets may be automatically merged between applications that form part of a suite or between a palette of components in a program builder. A number of different merge options are supported:

- Table of contents and index information may be appended one after the other
- Files can be sorted together
- Helpsets can be merged so common topics are not listed more than once
- Multiple databases can be searched during full text-search operations

Synchronization

JavaHelp software synchronizes the display of help system information between the content pane, table of contents, and indices through the use of scrolling and highlighting. In this way, users are always aware of their relative locations within the help system as well as related help topics.

Embeddable Help Windows

Embedded help windows provide easy access to specific help data where and when the user needs it. Help windows (e.g., viewer, help, or content) can be embedded directly into application interfaces, enabling developers and authors to assign help for specific components on the screen.

For example, when needing assistance in a particular area, the user might invoke embedded help rather than launching the entire help system. By providing an information specific window only when needed, help designers preserve screen real estate while offering key insights and options. Embedded windows can also provide access to the help system for the entire application.

Context-Sensitive Help

JavaHelp systems can also be used to present specific kinds of help at certain points within an application, based on the context in which the application is being used. For example, if a user is trying to perform an operation and clicks on help, the help system can provide information about how to perform that specific operation.

Flexible Data Packaging

Help information can be packaged in many different ways within a JavaHelp system. JavaHelp software information is typically packaged in JAR files, which can be customized and separated into client, server, and storage files to be retrieved and used as needed. The JavaHelp system extracts help information from JAR files for specific tasks. With flexible packaging, authors and developers enjoy many options for updating and deploying help system content.

CHAPTER 5

Using JavaHelp — Examples

JavaHelp software's flexibility offers help system authors many options for deploying or invoking it from within applications. JavaHelp is especially effective in networked applications, and can be launched independently of the application it supports. Other help systems do not offer authors and developers as much flexibility for as many different environments.

Deploying JavaHelp

Developers often must build a help system without knowing its final deployment environment. JavaHelp software enables help systems to be adaptable in presenting and displaying information. The following scenarios describe ways in which JavaHelp software can be used to present help information.

Standalone Applications

A Java standalone application is one that runs independent of a Web browser, accesses help data locally, and is installed and running on the same machine. Standalone applications can include just about anything — from a kiosk for online documentation to a standalone help system.

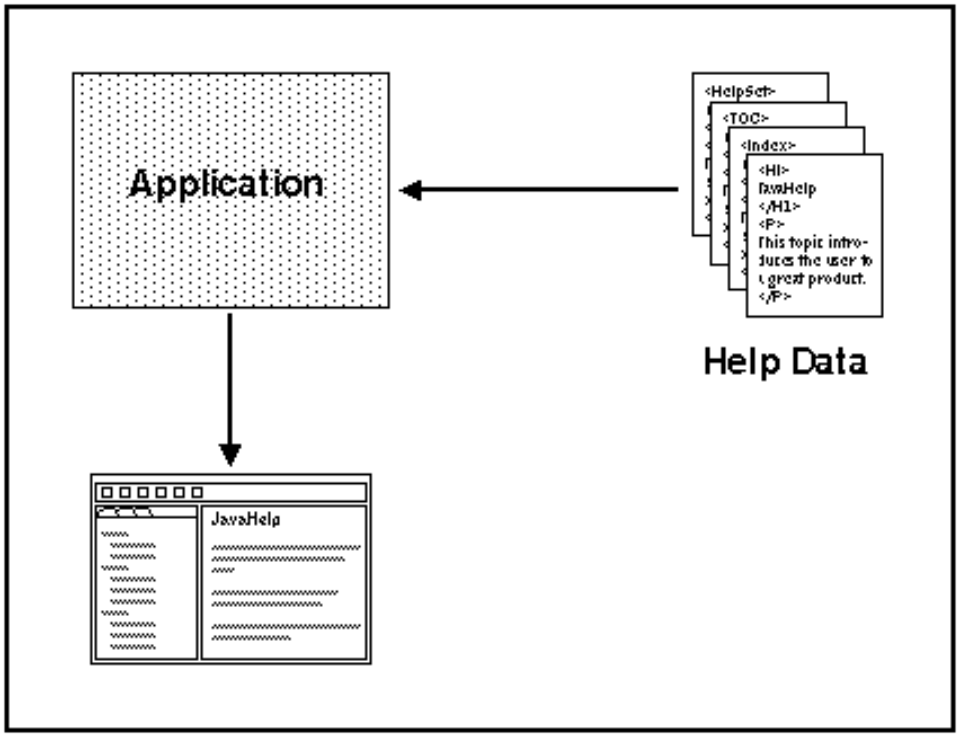

FIGURE 1 Separate JavaHelp Viewer Used with a Standalone Application

1. The application requests the creation of a JavaHelp instance
2. Help data is loaded from the network
3. The requested help topic is presented

Separate Viewer

Navigational and content information can be launched in a viewer separate from the application frame. This is the default implementation, and can be modified for help systems.

Embedded Viewer

Both navigational and content information may be embedded directly in application windows. This is achieved by embedding the JFC components that implement JavaHelp components directly into the application frame. Developers can also use these components to customize the presentation.

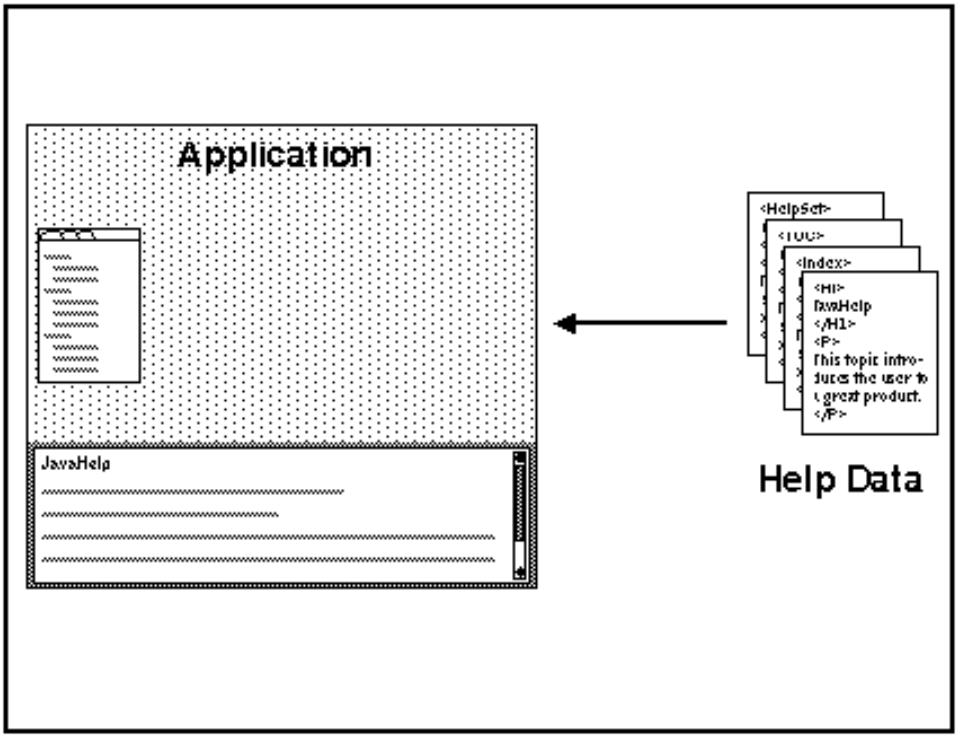

FIGURE 2 JavaHelp Viewer Embedded in an Application

In Figure 2, the content pane is embedded along the bottom of the application window, and the navigation pane is embedded in a different portion of the window. This enables the application to control information displayed by means other that the JavaHelp system navigators.

Invoking the JavaHelp System

JavaHelp software-based help systems can be invoked from within applications using a variety of methods, including menus and buttons, tooltips, and context-sensitive help.

Menus and Buttons

Online help is often invoked when a user chooses an item from a help menu or clicks on a help button in an application's graphical user interface. JavaHelp software provides a simple interface by which an application creates a help presentation by requesting a topic ID be displayed. The JavaHelp system associates the topic ID with the appropriate URL. IDs are mapped to URLs in a JavaHelp metadata file called the map file. For example, when coding a file chooser dialog box, a developer requests that the topic ID fc_help be displayed when the Help button at the bottom of the dialog box is clicked. In the map file, the ID fc_help is defined to be a file named *FileChooser.html* using the following syntax:

```
<mapid target = "fc.help" url = "ktml/help/filechooser.html"l>
```

By separating the specification of file names (or URLs) from the program code, developers and authors can control the information associated with the topic ID.

Tooltips

A tooltip is a brief message presented to the user when the cursor remains over a button for an interval longer than a given threshold. Although tooltip information could be included in the JavaHelp software data, it is usually delivered as part of the application, so it is located with the code. Tooltip functionality is provided as a component in the JFC.

Context-Sensitive Help

The JavaHelp system provides the ability to invoke online help for graphical components when the user activates context-sensitive help and then specifies the component in question. The ID associated with the component is displayed.

Networked Applications

The JavaHelp system transparently loads help data from corporate intranets and the Internet. When help data is accessed across a network, the location of the data is transparent to the application.

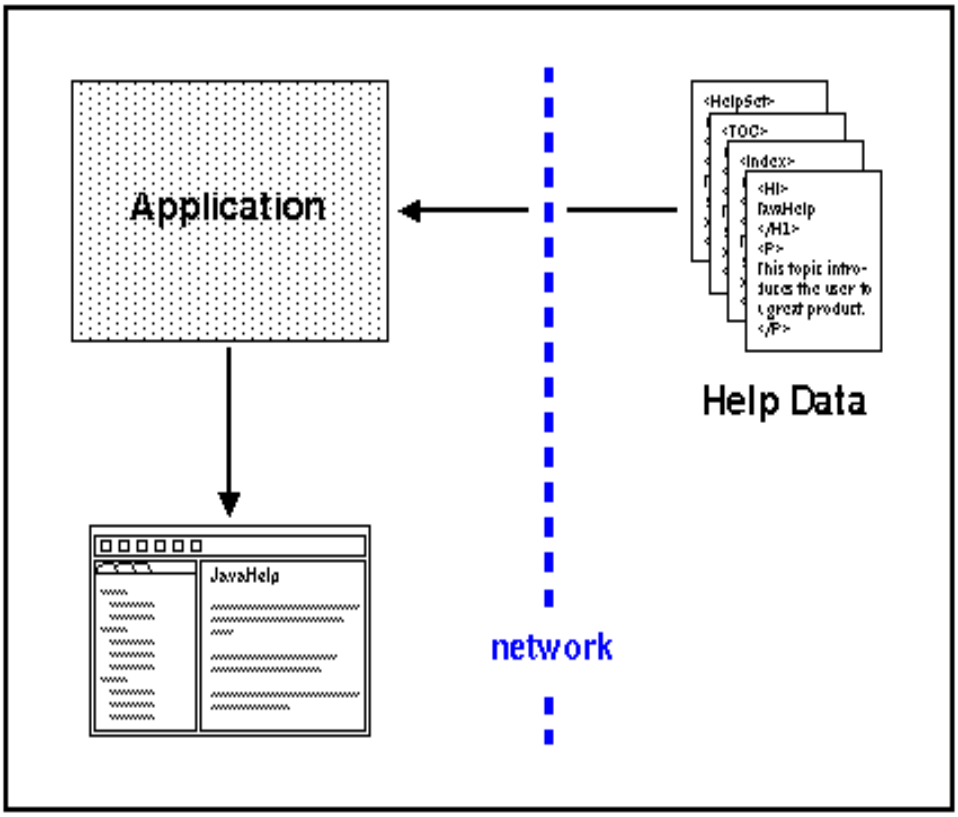

FIGURE 3 JavaHelp Viewer Accessing Networked Help Data

1. The application requests the creation of a JavaHelp instance
2. Help data is loaded from the network
3. The requested help topic is presented

Applets

Applets are applications delivered via a Web browser. They have unique deployment issues that are addressed by the JavaHelp system. The following three scenarios describe how the JavaHelp system is used from within browser-based applications. In these scenarios, an applet or some other triggering entity on an HTML page requests the JavaHelp system to display help information.

Applet Type 1

In this type of applet, JavaHelp software is already part of the user's browser and runs using the appropriate version of the Java Runtime Environment (JRE). The JRE may have been delivered with the browser, or it may have been downloaded by the client into the CLASSPATH. The implementation may use the JavaHelp software content pane, or it may use the HTML viewer that is part of the Web browser.

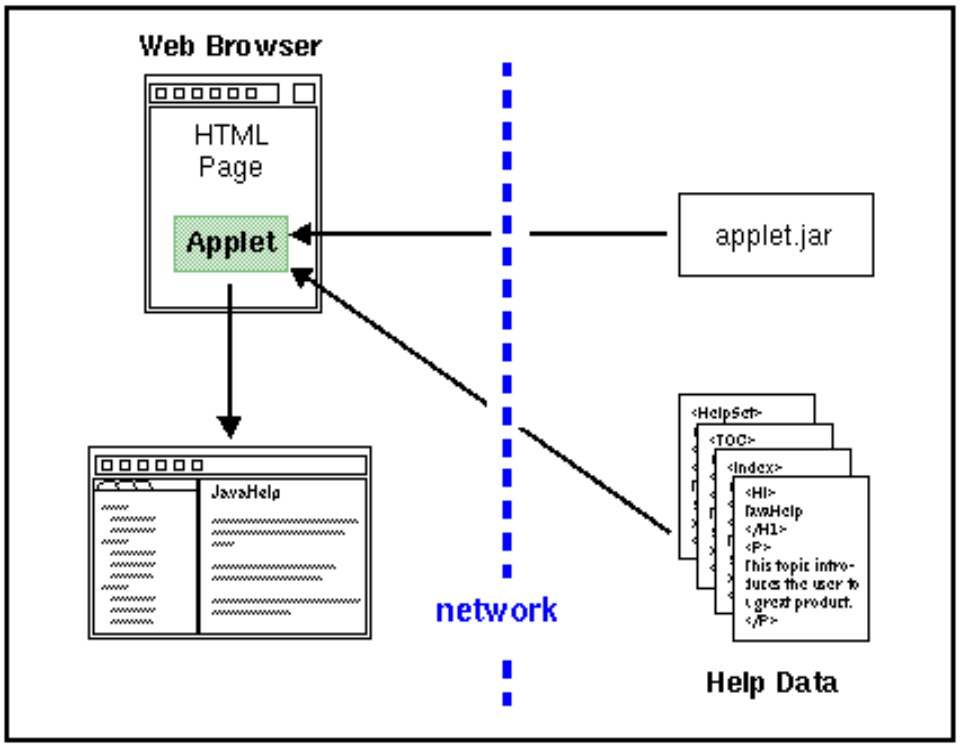

FIGURE 4 Type 1: JavaHelp Applet

1. The HTML page containing the applet tag is loaded into the browser
2. The applet is downloaded from the server and executed
3. The user requests help
4. The applet forwards the request to the JavaHelp system
5. Help data is downloaded from the server and displayed in the JavaHelp viewer or browser window

Applet Type 2

The second scenario occurs when the JavaHelp runtime version is part of the user's browser, but the help data is not part of the application and must be loaded over a network. JavaHelp software is a Java standard extension, and it is possible that it may not be in the CLASSPATH of a fully-compliant JDK 1.2 browser. If the HTML page refers to the standard JavaHelp implementation, the extension machinery automatically downloads the implementation and executes it. Since the JavaHelp system is quite small, this approach is often practical. Browsers may choose to provide other means for installing extensions downloaded through this mechanism.

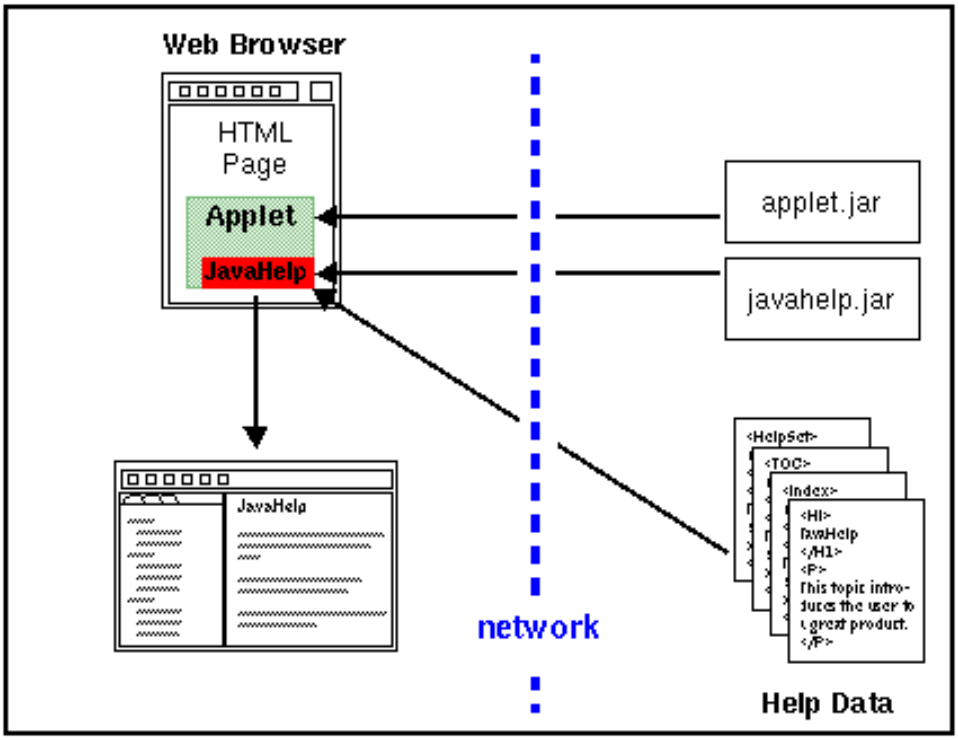

FIGURE 5 Type 2: JavaHelp Applet

1. The HTML page containing the applet tag is loaded into the browser
2. The applet is downloaded from the server and executed
3. JavaHelp classes are downloaded from the server
4. The user requests help
5. The applet forwards the request to the JavaHelp system

6. Help data is downloaded from the server and displayed in the JavaHelp viewer or browser window

---

Note – Steps 3 and 4 can be reversed.

Applet Type 3

The most complicated but versatile scenario is when the applet is downloaded to a browser environment without the appropriate JRE or JavaHelp system installed. In this case, the Java Plug-in can be used to download the required JRE and the JavaHelp standard extension classes. The Java Plug-in allows developers to specify a specific JRE on the HTML page that runs their applet. If the correct JRE is not present, the Java Plug-in downloads the correct JRE and installs it on the user's system. The new JRE is subsequently available to any applet that requires it. Because the JavaHelp system is a standard extension to the Java platform, the JavaHelp classes can be downloaded along with the JRE.

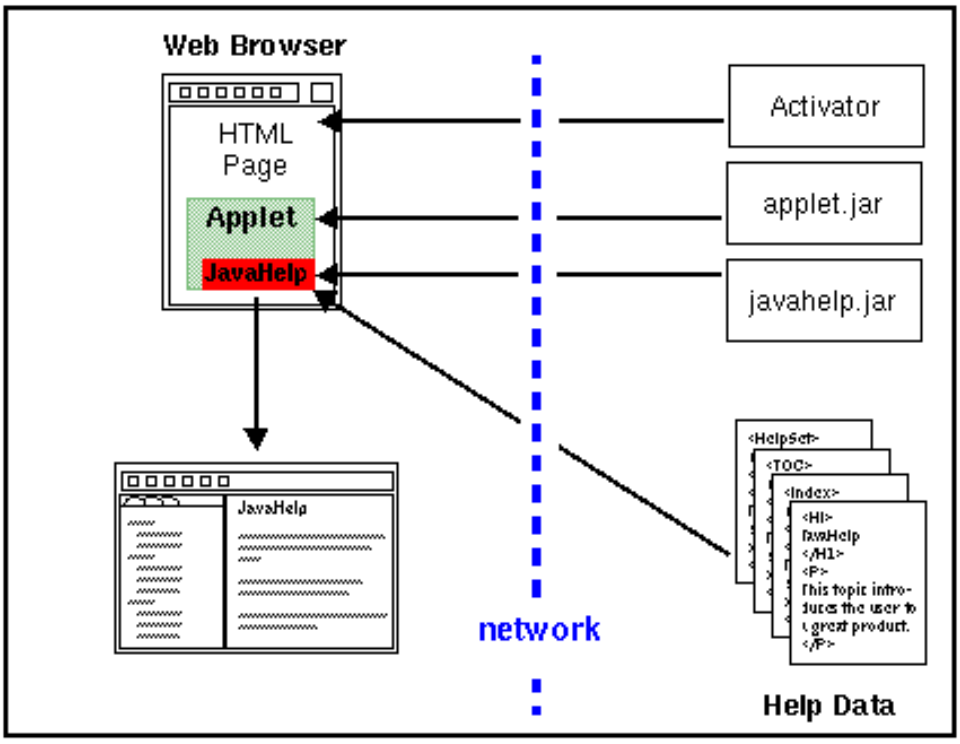

FIGURE 6 Type 3 JavaHelp Applet

1. The HTML page containing the applet tag is loaded into the browser
2. If required, the Java Plug-in is downloaded (it prompts the user to download the appropriate JRE and javahelp.jar)
3. JRE and JavaHelp classes are downloaded from the server
4. The applet is downloaded from the server and executed
5. The user requests help
6. The applet forwards the request to the JavaHelp system
7. Help data is downloaded from the server and displayed in the JavaHelp viewer or browser window

Full Text-Search Capabilities

The JavaHelp full text-search system is richly featured, compact, fast, and extremely flexible. The JavaHelp system includes a powerful search engine and search database indexer. Help authors can use the indexer to create a compact database that is distributed with the application's help data. When a user initiates a search, the search engine queries the database to determine matches. Alternative search engines can be substituted for the standard JavaHelp engine. The full text-search capability can be used in three different ways: standalone, client-side, and server side searches.

Standalone Searches

In a standalone search, all of the components (search engine, search database, and help content) are local to the application. From an implementation point of view, the standalone search is quite similar to the client search except that there is no need to cache the search data in memory or in local files. Help content files can be accessed locally or across a network.

1. A search is initiated
2. The search engine loads the database
3. The search engine scans the database and delivers hits to the application
4. The user or application chooses a hit
5. Content is loaded and displayed

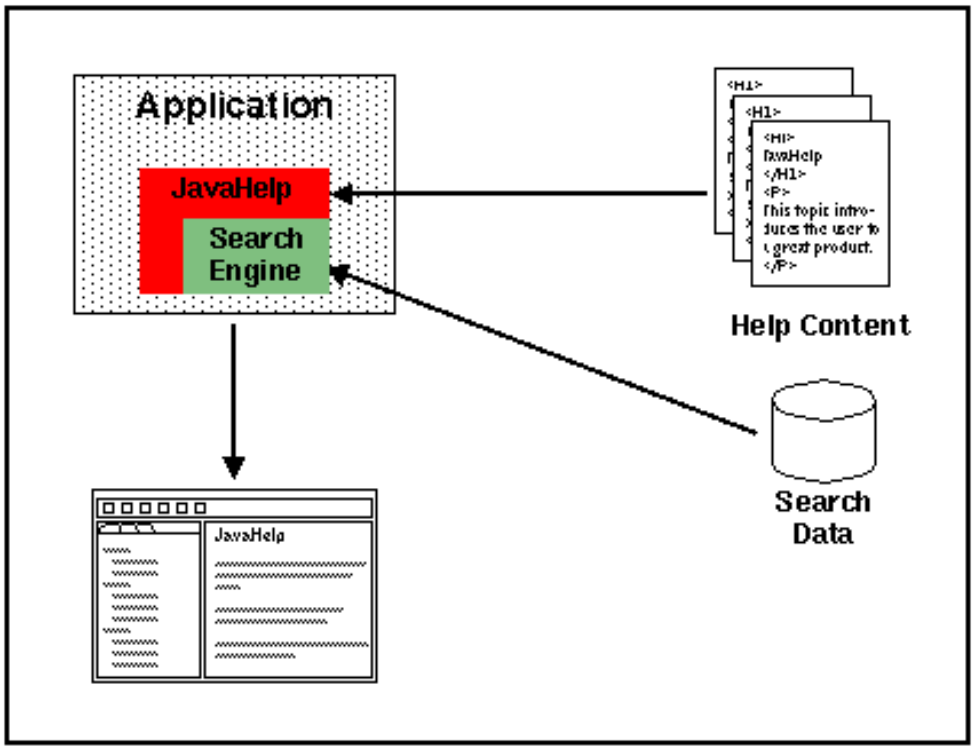

FIGURE 7 A Standalone Search

Client-Side Searches

In a client-side search, searching is done locally — literally on the client side — but the search data originates on a server. This arrangement is common with applets, where help data usually resides on the same server as the applet code. When a search is initiated, the data is downloaded from the server, which can take some time. Once downloaded, the database is read into the browser's memory and searched. Data can be kept in memory or in a temporary file on the client machine. The content files are downloaded only when they are presented. Once the database is downloaded, searches are quite fast.

1. A search is initiated
2. The search engine loads the database
3. The search engine scans the database and delivers hits to the application
4. The user or application chooses a hit
5. Content is loaded and displayed

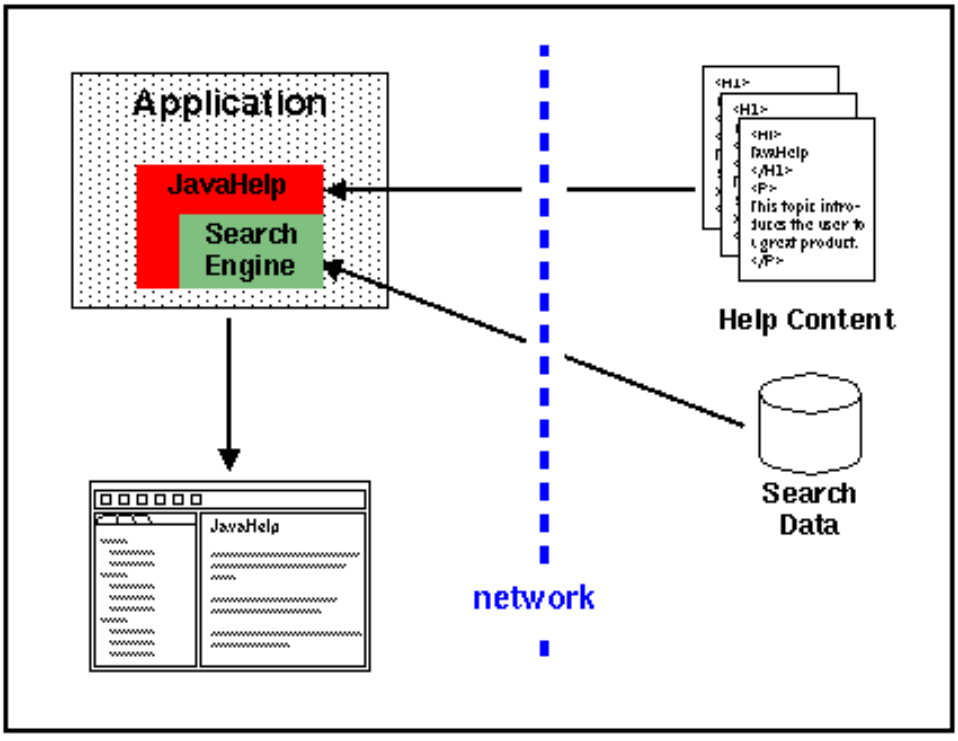

FIGURE 8 A Client-Side Search

Server-Side Searches

In a server-side search, both the search data and content files are located on the server, and only the results of the search are downloaded to the client. This option also works well for applets and Java servlets. It allows developers and authors to use alternate search engines (for example, Excite or Lycos) and can be quicker to start than downloading the search database, especially if the search engine is already running on the server.

1. A search is initiated
2. JavaHelp requests a search from a server
3. The server-side search engine scans the database and delivers hits to the application
4. The user or application chooses a hit
5. Content is loaded and displayed

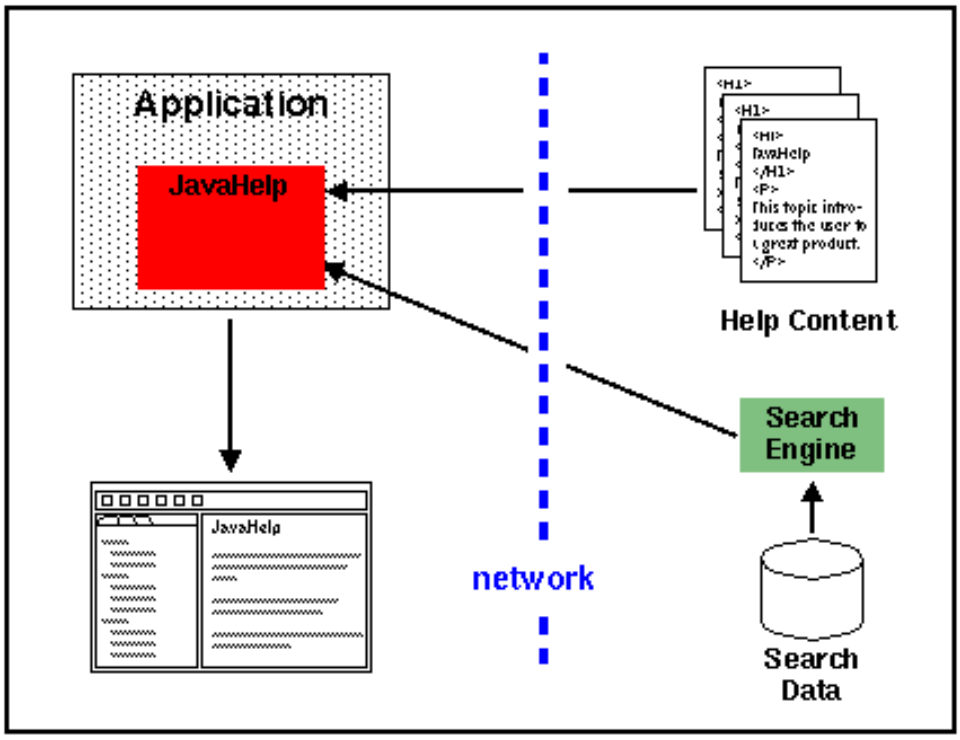

FIGURE 9 A Server-Side Search

Using JavaHelp with Components

Today's applications are often composed of interoperable components. Examples range from large applications like Netscape Navigator (with plug-ins) to applications where JavaBeans™ components are connected using JavaScript™ software or Visual Basic.

In the case of JavaBeans components, each component may be shipped with its own help data as shown in the following diagram.

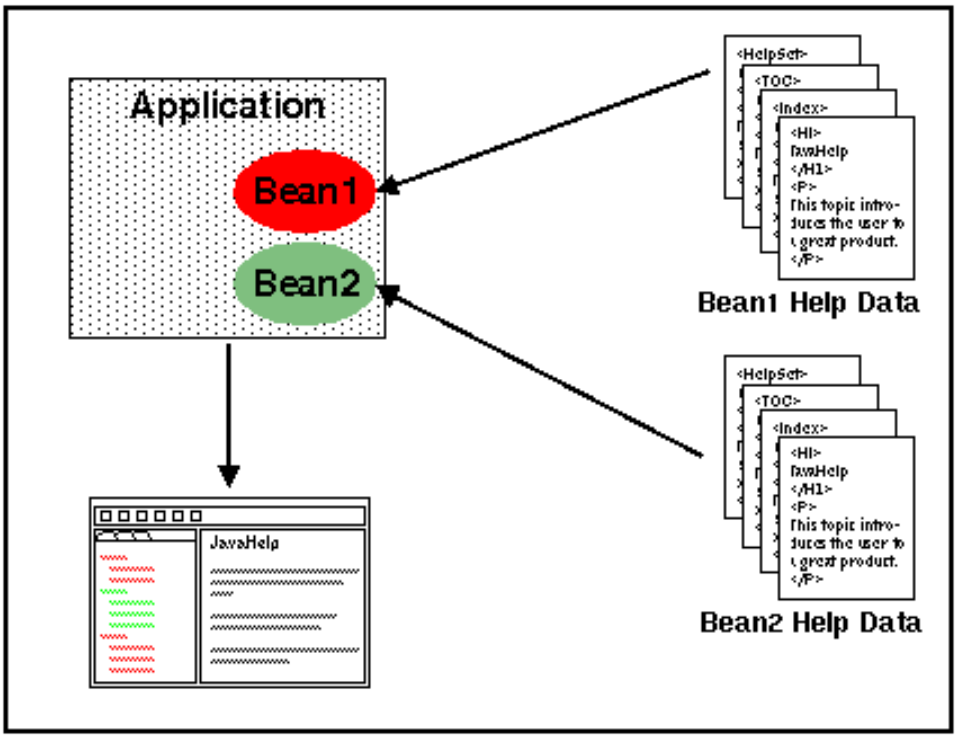

FIGURE 10 Using JavaHelp with Components

Help information from bean 1 and bean 2 are merged in the help viewer table of contents. The merge operation can be performed by the developer ahead of time, or completed when the application or JavaBeans component is installed by the user.

The JavaHelp system supports a number of different merge options. For instance, table of contents information for each component may be appended one after another, sorted together, or united so common topics are only listed once.

Launching JavaHelp Separately from an Application

In some environments, it is useful to separate the process that presents the help information from the application. For example, applications written in a language other than Java software (e.g., C, C++, and Visual Basic) can use the JavaHelp system to display online help when deployed on diverse computing platforms.

In another example, a suite of applications might be installed together or separately. In this case, the help server can be used to display help for the entire suite, rather than requiring each of the constituent applications to provide its own help system.

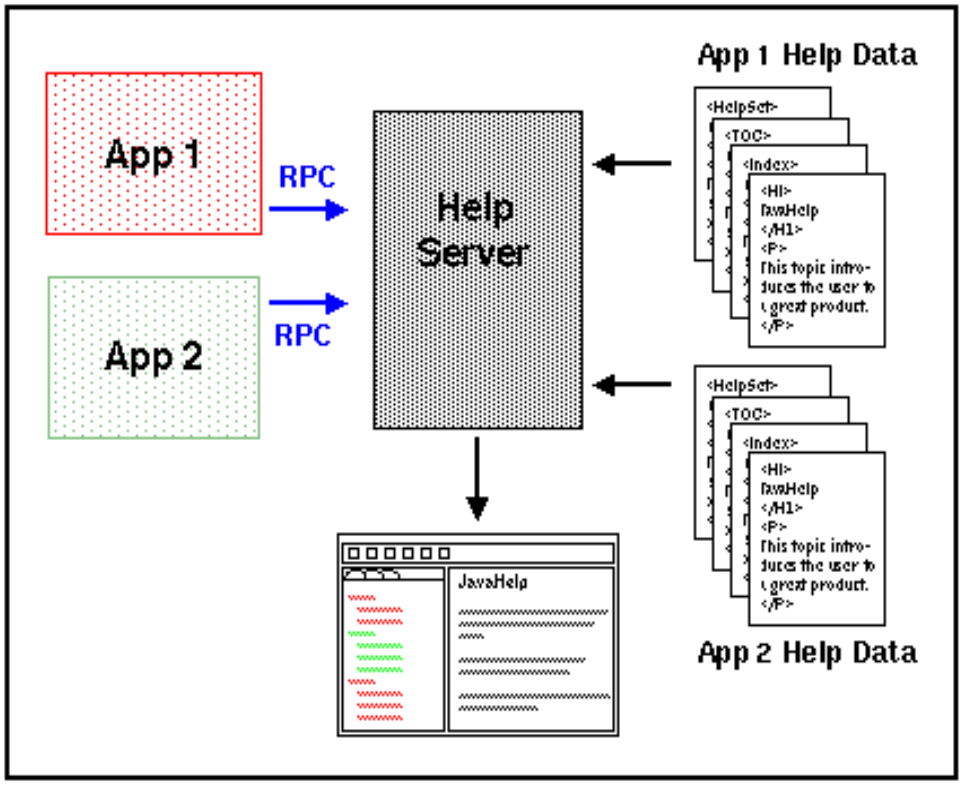

FIGURE 11 A JavaHelp Server

Applications that are not based on Java software make requests to a JavaHelp process residing on a help server via a remote procedure call (RPC) mechanism. The RPC may be wrapped in a library and be made invisible to the application developer.

CHAPTER 6

Summary

The JavaHelp API is the choice for developers and help authors who need to provide application help for multiple computing platforms, Web-based applications, and online documentation. As the only true cross-platform solution, JavaHelp software enables help systems to be written and used by virtually any user on any platform with any browser. As a 100% Pure Java API, JavaHelp software offers a flexible, rich environment that can be customized or extended to meet the help system needs of a wide range of applications. With strong authoring tool support and the ability to simplify localization of help systems, JavaHelp software allows developers and authors to add value to applications without an enormous investment in programming and development.

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303

1 (800) 786.7638
1.512.434.1511

http://java.sun.com

October 1999