A Help Facility for X Window System Applications

Mark Newsome
Department of Computer Science and Engineering
Auburn University, AL 36849
mnewsome@eng.auburn.edu

Abstract

Xhelp *is an interactive help facility for X-based applications, providing a friendly, consistent help interface across a variety of applications.* ***Xhelp*** *may be invoked as a standalone utility for tutorials or general system information. Its programmatic interface also reduces the coding effort required to integrate the help facility for automatic invocation by application software.*

Keywords: online help, X Window System, user interface design

Introduction

Software manuals are beneficial only when used. The problem with printed texts is that their physical characteristics can impede their use. Often, manuals are not located close to the computer or may need to be retrieved from another person or building. On a desktop already occupied by a keyboard, mouse, monitor, and system unit, manipulating manuals may be clumsy. It can also be quite tedious to search pages of text for a key word or phrase. Finally, printed examples (e.g. program code fragments) must be re-typed in order to insert them into a text editor.

Online access to software documentation offers several clear advantages over printed manuals:

- Online documentation reduces desktop clutter.
- Information is available whenever the software itself is available.
- Online manuals reduce paper consumption.
- Because online manuals are maintained electronically, safeguards against inadvertent use of out-of-date material can be implemented.
- Searching or indexing capabilities allow retrieval of specific information quickly.
- Information from the manual can be copied into other applications with conventional cut-and-paste operations.

Since onscreen manuals offer clear advantages over printed media, why are they not widely used? Poor display technology has been cited as the primary reason [1]. Text displayed on a low resolution monitor is not as readable as text printed on paper. Moreover, small screens require that the user switch back and forth between displays. Recent advances in video electronics, however, have made the concept of online documentation more appealing. High resolution monitors provide displays rivaling

the eye-appeal of printed text. Larger screen areas permit the use of multiple windows so the viewer can refer to the help display without blocking the application interface.

The lack of a standard, easily accessible help facility for the X Window System prompted development of **xhelp**. Written under the direction of Cherri M. Pancake at Auburn University[1], the program incorporates features benefiting both users and developers of X applications. **Xhelp** opens a scrollable help window with paging, browsing, and text search capabilities (Figure 1). The program can be used as a standalone utility for tutorials or general system information. Alternatively, other application programs may communicate help requests to **xhelp** through a software interface. A built-in editor makes it easy for application developers to create or modify help text.

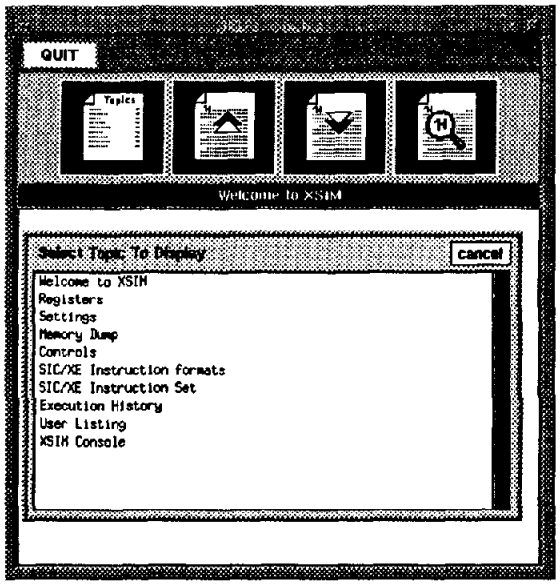

Figure 1. Xhelp User Interface.

This paper describes the **xhelp** system. First, the facilities for online assistance and documentation provided by commercially available tools are surveyed. This is followed by a description of **xhelp** and its use as a standalone X client or as an integrated help facility for other X applications. A concluding section addresses the current status of **xhelp** and its availability through MIT's distribution of X11R5.

Other Online Help Facilities

Help facilities in application software differ in purpose, invocation, presentation, and features. The help engine is either built into the application or implemented as an external program. For text-based software, the most common form of help is "reminder screens", providing a quick reference of function/control keys definitions and command summaries. WordPerfect's help screen is an example of this type of help (Figure 2).

Features (H)	WordPerfect Key	Keystrokes
H-Zone	Format	Shft-F8,1,1
Hanging Indent	Indent	F4,Shft-Tab
Hard Hyphen	Hard Hyphen	Home,-
Hard Page Break	Hard Page	Ctrl-Enter
Hard Return Display Character	Setup	Shft-F1,2,6,4
Hard Space	Space Bar	Home,Space
Hard Tab	Tab	Home,Tab
Headers	Format	Shft-F8,2,3
Help	Help	F3
Help, Printer	Print	Shft-F7,s,6
Hidden Codes	Reveal Codes	Alt-F3
Hide Document Comments	Setup	Shft-F1,2,6,2
Hyphen Character	Hyphen	-
Hyphen, Hard	Hard Hyphen	Home, -
Hyphen, Soft	Soft Hyphen	Ctrl, -
Hyphenation	Format	Shft-F8,1,1
Hyphenation Dictionaries	Setup	Shft-F1,3,6
Hyphenation Files	Setup	Shft-F1,6,3
Hyphenation Prompt	Setup	Shft-F1,3,7
Hyphenation Rules	Setup	Shft-F1,3,6

Figure 2. WordPerfect Help.

Help modes/levels are employed in some applications to control the occurrence, type, or level of help displayed. Usually, a portion of the screen is dedicated to help displays that change as the user traverses the software. Word Star's settable help level allows users to match the amount of help provided to their levels of expertise. IDE's Software through Pictures employs a help switch to toggle help displays on and off.

[1] A portion of the work was funded under an Auburn University Teaching-Grant-in-Aid.

Application-independent help programs, such as those found in most operating system environments, provide assistance with command syntax and features. Another example is the UNIX **man** utility, which displays manpages (manual pages) for applications, utilities, and library functions. Manpages, retrieved by keyword or name, are typically composed of a brief description, options, a list of known bugs, and author credits for the program.

Some help facilities offer *context-sensitive* assistance reflecting the user's current location in the software. The programming environment used in Borland's Turbo languages make effective use of this feature. Placing the cursor over a library function and pressing Ctrl-F1 pops up a help window with a function description and a list of required parameters, followed by an example. This lookup capability virtually eliminates the need for hardcopy reference manuals.

Recently, help facilities and development tools were released for several of the most popular graphical environments. In Windows 3.0, Microsoft introduced the MS Help System [2]. This is a hypertext based help facility, providing push-button controls for browsing, word search, and help file retrieval (Figure 3). Selecting an embedded hyperlink, represented as an underlined word/graphic, advances the help display to the related material. Development tools are provided with the Microsoft Software Development Toolkit (SDK).

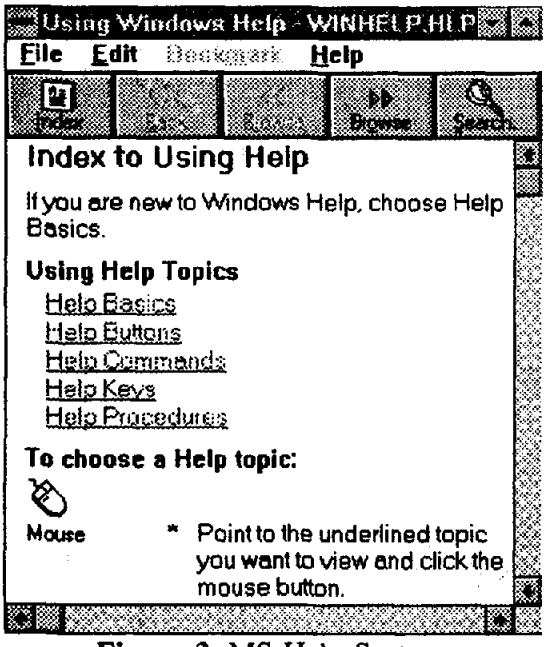

Figure 3. MS Help System.

Similarly, Apple released "Help Balloons" as part of the System 7 version of the Macintosh Operating System [3]. Help balloons look like the dialog bubbles used in comic strips (Figure 4), with tips pointing to the objects they annotate. Help balloons contain descriptive text or pictures for objects such as scroll bars, menus, and buttons. For example, when the cursor is moved over a menu item, a help balloon points to the item, explaining its function. Although the balloons are visually appealing, they are not well suited for in-depth explanations. The Macintosh Operating System manual furnishes information for developers on providing application access to the help balloons.

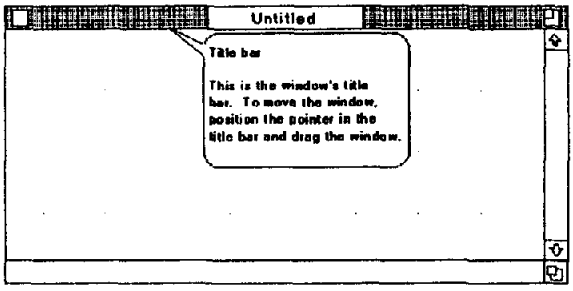

Figure 4. Mac Help Balloon.

Unfortunately, no standard help facility has emerged for users of the X Window System. While several help systems exist, most are proprietary -- that is, requiring expensive licensing fees. The only public-domain help system is **xman**, a windowed version of **man** (Figure 5). **Xman** retrieves manpages by keyword or by name, and displays them on a paper-like window.

Designed strictly as a standalone application, **xman** does not provide an application accessible interface.

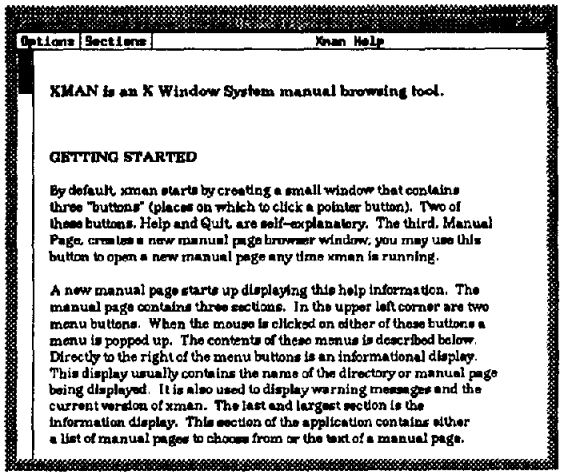

Figure 5. Xman.

IBM's InfoExplorer [4], a CD-ROM based information retrieval system, provides hypertext access across all application, installation, and hardware manuals supplied with the IBM RS/6000 series of equipment (Figure 6). InfoExplorer provides a variety of retrieval and navigation techniques, including a task index, word search, history mechanisms, and several types of topic lists. Like **xman**, InfoExplorer does not permit application access, but only functions as a standalone program. Since InfoExplorer is a proprietary program, it is not freely open for outside developers to use with their applications.

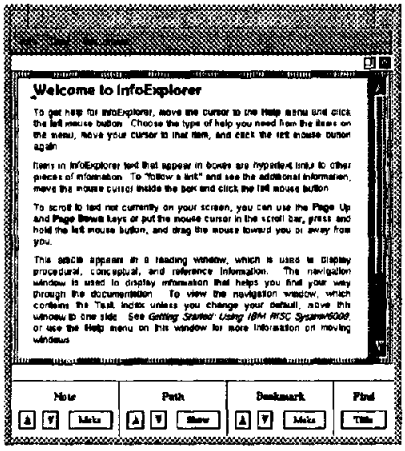

Figure 6. IBM InfoExplorer.

The xhelp System

The lack of a standard, easily accessible help facility for the X Window System prompted development of **xhelp**. The program performs double duty as both a help viewing system and a help development system. *User mode*, accessible via the command-line or through the programmatic interface, supports indexing, browsing, and searching of the helpfile by end users. *Developer* mode provides the additional editing and file storage capabilities necessary to create and maintain helpfiles for individual applications.

The **xhelp** interface (Figures 1, 7, and 8) consists of three areas: a menu bar, push button controls, and a scrollable text window. The number of items on the menu bar varies according to the mode in which **xhelp** is invoked. In user mode, the bar consists solely of a *quit* button, used to exit help. In developer mode, *file* and *edit* menus are activated to support helpfile development.

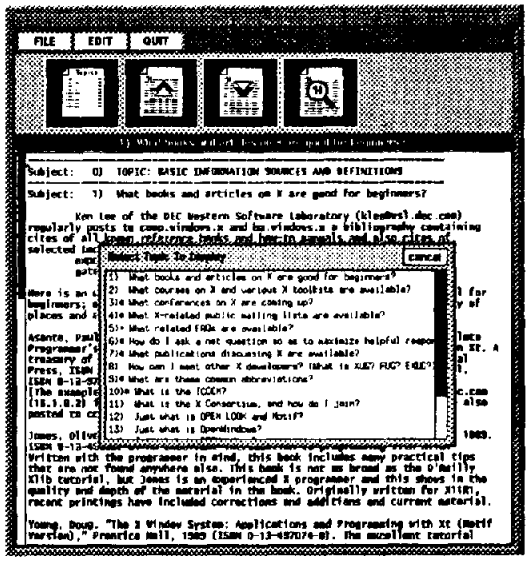

Figure 7. X FAQ (Frequently Asked Questions).

Below the menu bar is a panel containing four iconic push-buttons, used to operate the help facility. The left button pops up a list of available help topics, which can be selected for display. The next two buttons, marked with up and down arrows, are used for browsing through help topics one at a time. The "magnifying glass" button on the right provides support for keyword or phrase searches.

The main help window displays another title bar containing the name of the current help topic, as well as a scrollable text area. When the user chooses one of the topic browse controls, a secondary window pops up, offering a selectable list of topics on which information is available. In user mode, neither the title bar nor the help text may be altered, although the text may be copied for pasting into other applications. Developer mode, however, accommodates the full variety of standard editing features supported for the Athena **text** widget. [5]

Xhelp was designed so that the appearance of the user interface may be customized by the end user or the applications developer, using entries in the X Window System resource database. [6] The **xhelp** structure and the Athena widget documentation [5] provide important naming and attribute information needed for resource database entries. Customizable features include key bindings, colors, fonts, component placement, and size attributes.

Developing help files

Helpfiles are organized as a series of helptopics, each consisting of a *topicname*, *topicid*, and *helptext*. The first entry of a helpfile is reserved for the application introduction. Application *helptopics* may include tutorials, tables, templates, and glossaries, in addition to interface usage topics and explanations of application features.

In developer mode, two additional menus, *file* and *edit*, appear on the menu bar. The *file* menu provides selections to create a **new** help file, **open** an exiting one, **save**, and display info (author, version, and copyright) **about** the help facility. The *edit* menu lets the developer **add**, **delete** topics, and list help topics (with **ids**).

The first step in creating a helpfile is organizing the help information into topics, or context-strings. For example, possible topics for a text-editor include "deleting a line", "inserting a line", and "saving a file". A *helpfile* is created by selecting **new** on the file menu. At this point, the first help topic -- the application introduction -- is entered on the initially blank help screen. Additional help topics then may be defined using the topic entry dialog which appears when **add** is selected from the *edit* menu. Help text may be entered manually in the help window, or copied from another application using conventional cut-and-paste operations.

When the editing session is complete, the *helpfile* is written to a text-file using the *save* selection (*file* menu). To avoid inadvertent file loss, a verification dialog appears if the *quit* button is pressed before saving.

Programmatic Interface

Xhelp may be run from the command-line for standalone operation in either *user* or *developer* mode. A programmatic interface, however, offers a simple and convenient means of embedding help access within new or previously developed applications. When an X client application issues a help request through the predefined interface, a transparent mechanism automatically invokes the **xhelp** facility.

Inclusion of the header file "xhelp.h" provides access to the programmatic interface. Within the application program, **xhelp** is invoked using the following convenience function:

void XhCallXHelp(Widget toplevel; String helpfile; String topicid)

The first parameter is the application's *toplevel* widget. The second, *helpfile*, is a fully qualified filename, including path and an optional ".hlp" extension. The *topicid* is a unique mnemonic assigned by the developer to the help topic. (For convenience in coding help requests, the **list ids** selection available on the edit menu displays a list of topic ids.) Calls to **xhelp** with undefined *topicids* default to the introductory help topic.

In the application interface, help buttons may be assigned to always bring up the same help text, or to bring up specific help screens based on the user's location in the software. For instance, a special button can be reserved to display interface help. An "about" button could serve to present information (author, copyright, version) about the software. Other buttons could be defined to display information about the current operation being performed.

The software interface is able to detect whether or not **xhelp** is already active, and will instantiate a new client when necessary. Thus, a single **xhelp** window can be used to service all client applications, displaying the *helptopic* of the most recent request.

Status and Distribution

For the past year and a half, **xhelp** has furnished online support for the **xsim** computer simulator [8] in the systems programming laboratory at Auburn (Figure 8). To assist users not familiar with **xsim**, a button marked "??" invokes **xhelp** to display information on the simulator. Window-specific help is obtained by pressing the black and white title bars. The "REGISTERS" titlebar, for example, reveals information about register usage, changing display representation, and altering register values. Tutorials on base conversion and debugging are available through the **xhelp** index.

In addition to its role in providing online documentation, **xhelp** turned out to be convenient as a class bulletin board facility. Using the help index, students could find postings on assignment hints, program fragments, and announcements. **Xhelp's** cut-and-paste feature offered an easy means to copy program fragments from the help window, saving time and eliminating typing mistakes.

In standalone operation, **xhelp** has been used as an indexed retrieval system for the X FAQ (Frequently Asked Questions) list (Figure 7). The list of more than 120 questions and answers were easily pasted into **xhelp**. Questions were transformed into help topics; answers became help text. **Xhelp**'s search feature makes it convenient to find keywords within the list.

The **xhelp** system is distributed with the user contribution portion of the X11R5 release of the X Window System [7], distributed free of charge by MIT[2]. Written in C, **xhelp** employs widgets from the Athena widget set, also provided with the X distribution. To simplify installation, an **imake** file is included with the software (README file outlines installation customization). The **xhelp** system has been ported to several architectures including the IBM RS/6000, Sun SparcStation (monochrome and color), Sun 3/50, Sony Workstation, and IBM 3090 Supercomputer.

Xhelp makes it possible to add context-sensitive help easily and consistently across a range of X-based applications. Enhancements under consideration for future versions include compression of help files, graphics support, and hyperlink access.

[2] Obtained by anonymous ftp from export.lcs.mit.edu.

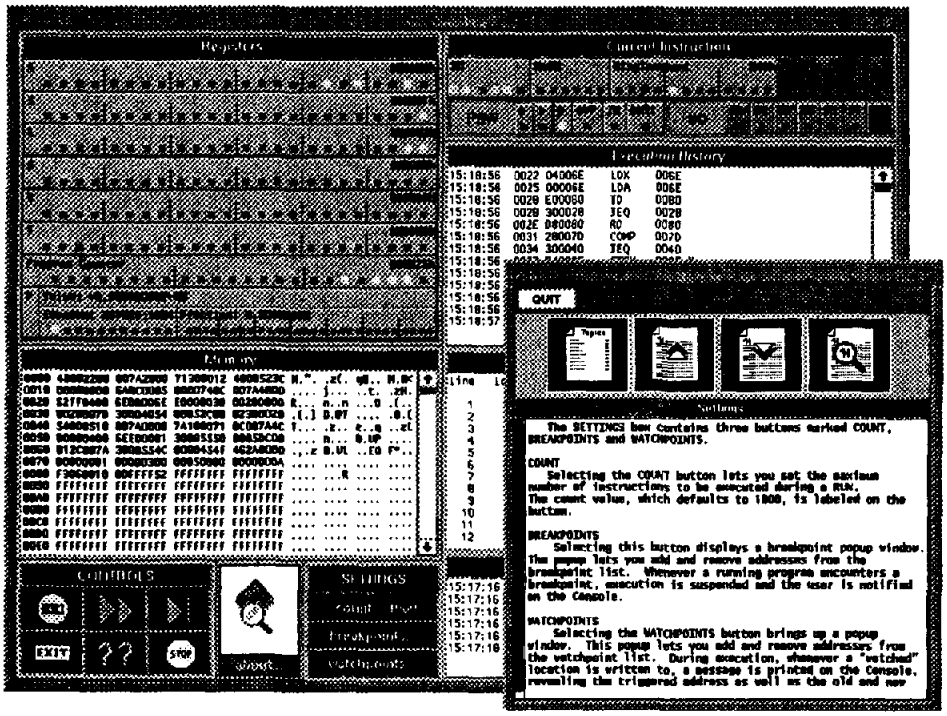

Figure 8. Xhelp supporting xsim.

References

[1] Shneiderman, Ben. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Addison-Wesley, Reading, MA, 1987.

[2] Adler, Marc. "Adding Hypertext-based Help to Your Application Using the Microsoft Help System." *Microsoft Systems Journal*, vol. 5, no. 3, May 1990, pp. 325-336.

[3] Danuloff, Craig and Aileen Abernathy. "Things You Need to Know About System 7.0," *MacUser*, vol. 7, no. 6, June 1991, pp. 70-96.

[4] International Business Machines. *Getting Started: Using RISC System/6000*. Austin, TX, pp. 4.1-4.27.

[5] Asente, Paul J. and Ralph R. Swick. *X Window System Toolkit*. Bedford, MA, Digital Press, 1990.

[6] Quercia, Valerie and Tom O'Reilly. *X Window System User's Guide*. Sebastapol, CA, O'Reilly and Associates, 1990.

[7] Scheiffer, Robert and James Gettys. *X Window System: Complete Reference to Xlib, X Protocol, ICCM, XLFD*, Second Edition, Bedford MA, Digital Press, 1990.

[8] Newsome, Mark R. and Cherri M. Pancake. "A Graphical Computer Simulator for Systems Programming Courses," *Proceedings of the SIGCSE Technical Symposium*, Kansas City, MO, 1992.