

ASNA

“Creating Help Files” for Smarties



*A Step-By-Step Tutorial to assist you
in Creating and Connecting Help Files
to Windows and Web Applications*

Learn about Help Files and their
components and formats.

Create an entire help file using
Robohelp by Ehelp Corporation.

Connect the Help File to an AVR
Windows application.

Convert the Help File to WebHelp
and connect a Web application to a
Webhelp help file and context-
sensitive help in a separate window.

By Julie O'Brien

ASNA

“Creating Help Files” for Smarties

Information in this document is subject to change without notice. Names and data used in examples are fictitious unless otherwise noted.

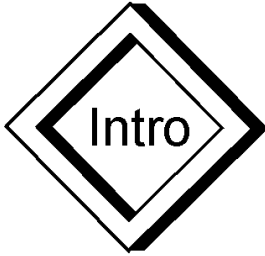
No component of ASNA Visual RPG for Smarties may be reproduced, disassembled, transmitted, transcribed, stored in a retrieval system, or translated into any language in any form without the written permission of ASNA (Amalgamated Software of North America).

Released for ASNAPalooza 2001 March, 2001

Contents

Introduction: Getting Started and About Help Files.....	1
Using the Tutorial	1
How to use this Tutorial.....	1
Source Files.....	1
Saving Your Work	2
Manual Conventions	2
What Do you Want to Do?.....	2
About Help Files	3
What is a Windows Help File?.....	3
Why Create a Help File?.....	3
How can you use Help?	3
Types of Help Formats.....	4
Considerations Prior to Creating a Help File	7
Where Do I Start?	7
Which Help Authoring Package?.....	7
Help File Components.....	8
Topics.....	9
Images and Multimedia.....	10
Windows	10
Links	11
Table of Contents	12
Index	12
Macros	13
Step 1: Creating a Help File.....	15
Creating a New Help Project.....	17
Getting Around in RoboHelp	19
RoboHelp Explorer (Left Side).....	20
Microsoft Word (Right Side)	20
Inserting an Existing File	21
Saving the File	22
Creating an Index	22
Properties of a Help Topic	23
Adding New Topics	25
Creating Links.....	26
Creating Context-Sensitive Help.....	27
Adding HTML Topics.....	29
Compiling the Help Project.....	31
Creating a “Contents” Tab	32
Adding a Graphic	33
Inserting Multimedia Files	34
Inserting a Wave File	34
Window Settings	35
Changing the Background Color of a Popup.....	37
Setting Project Settings	39
Compiling the Help Project.....	41
Creating a Map File.....	42

What is WebHelp?	45
Generating WebHelp.....	46
Step 2: Connecting a Help File to an AVR application	51
Calling Help from a Help Menu.....	53
Calling an HTML File Locally or on the Web.....	54
Calling Help using a Help Button	54
Using the Map Number Parameter.....	55
Using the Help Topic ID Parameter.....	56
Using CommonDialog to Display a Help Topic by Keyword	56
Winhlp32 Parameters.....	57
Creating Context-Sensitive Help from an Application.....	58
Specifying the Help File for the Project.....	58
Assigning Context-Sensitive Help in the Application	59
Using the WhatsThisHelpID Button and Property.....	63
Step 3: Connecting WebHelp to a Web Application	67
What We Will Cover in Step 3.....	68
What is WebHelp?	68
Calling the Entire WebHelp file from a Help Button.....	69
Displaying the Entire Help File in a Separate Window.....	70
Displaying Context-Sensitive Help in a Separate Window.....	71
Index	75



Introduction: Getting Started and About Help Files

Using the Tutorial

This tutorial uses RoboHelp, AVR/Caviar and FrontPage 2000. Some prior experience in these applications may be helpful, but not necessary.

How to use this Tutorial

This tutorial is designed as a “self-learning” tool and includes 3 additional chapters. Each chapter is a “step” that can be performed independently, and does not rely upon the completion of the previous step. You can merely use the source files needed that are included on the CD.

At the beginning of each chapter, you will find:

- A summary of the topics that will be covered in that step.
- The approximate time to complete that step.
- A visual representation of what the application will look like at the completion of that step.

Source Files

The completed source files for each of the tutorial steps can be found in the **\Session 05 – Creating Help Files** and **\Session 10 – Connecting Help Files to Apps** folders as listed below.

- The completed **help file** that is created in Step 1 can be found in **\Session 05 – Creating Help Files\Examples\Help_File\Source Files**.
- The completed **WebHelp** file that is created in Step 1 can be found in **\Session 10 – Connecting Help Files to Apps\Examples\Web_Help**.
- The completed **AVR application** that is created in Step 2 can be found in **\Session 10 – Connecting Help Files to Apps\Examples\AVR_App\Completed App**.
- The completed **Web application** that is created in Step 3 can be found in **\Session 10 – Connecting Help Files to Apps\Examples\Web_App\Completed App**.


Saving Your Work

When you save your work for a particular step, place it in a separate folder called **Mine** (or whatever you wish) instead of saving it directly onto the original file.

This way, the original source files do not get overwritten, and you can compare your application to that of the tutorial.

Manual Conventions


The following is a list of conventions that are used throughout this tutorial representing particular functions.


- A  beside a headline indicates a step-by-step process that you are to perform.
- Numbered lists (1, 2, 3, and so on) indicate steps that you should follow.
- Menu options, buttons, controls and keys that you are to select, along with words that you are to type will appear in **boldface**.


What Do you Want to Do?


This manual is both a reference guide and a step-by-step tutorial. The Introduction chapter is strictly for explanation and reference, and Steps 1-3 are step-by-step tutorials.

Note: You do not need to perform Step 1 in order to complete Steps 2 and 3, as the help file that is created is already included with the AVR application folder within \Session 10 – Connecting Help Files to Apps.

 For information about **Help Files and components** – refer to information later in this chapter.

 For the step-by-step tutorial to create a Help file using RoboHelp see **Step 1 – Creating a Help File**.

 For the step-by-step tutorial to connect an existing Help file to an AVR application see **Step 2 – Connecting a Help File to an AVR application**.

 For the step-by-step tutorial to connect a Webhelp file to a Web application see **Step 3 – Connecting a Help File to a Web application**.

About Help Files

Help Files are an integral part of a professional application. They provide a quick and informative way of presenting information to users. They are similar to printed documentation, but have additional benefits, such as hypertext links, allowing users to jump from one topic to another, or audio-visual capability that print media cannot provide.

A Help File has an .HLP extension and runs under the Microsoft Windows Help program **WINHELP** (Windows 3.1, 95 or NT) or **WINHLP32** (Windows 95/NT only). Users only need Windows and the .HLP File.

This chapter contains information on what a help file is and what components it has. Note that this step merely contains overview information, and there is nothing to do or create in this step. It is not necessary to read this chapter before continuing with Steps 1, 2 and 3. Continue with Step 1 to create a RoboHelp Help file, or continue with Step 2 to connect the existing help file to the AVR application, or Step 3 to connect the Webhelp file to a web application.

What is a Windows Help File?

- A Help file is online information containing groups of related topics connected by hypertext links.
- The user is in control - you can quickly and easily find information and move around at will.
- You can access the information quickly, when you need it without leaving your computer.
- Help files can contain graphics, multimedia, tutorials, links to other help files or to HTML or ASP pages.
- Help files can be in several different formats (.hlp, .chm, .htm).

Why Create a Help File?

- Provides users with instant, accurate, and up-to-date assistance and information on using your product.
- Increases knowledge, comfort level and confidence of the customer - just knowing the information is “there” .
- As knowledge increases - phone support calls decrease.
- Puts user in control - loves flexibility and addition of graphics, color, sounds, videos, buttons, and more.
- With added technology and features of a help file - the user “expects” it with a software application.

How can you use Help?

- Software Application assistance. This is probably the most familiar use of Help.
- Tutorials and computer based training. This type of Help can teach general techniques, features, or capabilities of a product or application.
- Manuals and handbooks. User guides, employee manuals, policy and procedure books, human resources handbooks, standards and practices, and more.



- Online books. Online books can be interactive. Great for Sales and Marketing materials.
- Catalogs and price lists.

Types of Help Formats

- WinHelp - 3.1 systems
- WinHelp - 95/98/NT systems
- Microsoft HTML Help
- Cross-platform Help
- Netscape NetHelp
- Intranet help
- Other Help formats
- Windows CE
- Java-based help

WinHelp - 3.1

- WinHelp - is the original help format that runs on a Microsoft Windows operating system.
- Uses a WinHelp compiler (HCP, HCW or HC31.EXE) to process the Help file.
- Contains an .HLP extension.
- Uses WINHELP.EXE engine to run, which automatically comes with the Windows Operating System.
- Can also run this file on Windows 95/NT systems.

Issues:

- Windows 3.1, or 16 bit applications are going away.
- Works on Windows platforms only.
- Doesn't allow "Books and Pages" .
- Limited features and capabilities.
- Can't link to a web site or an HTML page.
- Does not support multimedia.

WinHelp - 95/98 and NT

- WinHelp - the original help format with the addition of a contents tab and a new interface containing books and pages.
- Uses a WinHelp compiler (HCP, HCW or HCRTF.EXE) to process the Help file.
- Generates two files, an .HLP and a .CNT extension.
- Uses WINHLP32.EXE engine to run, which automatically comes with the Windows Operating System.

Issues:

- Will not run on Windows 3.1 systems.
- Microsoft is establishing HTML Help as the standard.
- Works on Windows platforms only.

Strengths:

- Will display using Winhlp32 on 32-bit Windows systems.
- Can link to a web site or an HTML/ASP page.
- Depending on authoring package used, can automatically convert to compiled (.CHM) or uncompiled HTML.

HTML Help

- HTML Help - developed by Microsoft in August 1997 is a completely new online Help standard for Windows 98, Windows NT 5, and future Microsoft applications.
- Contains individual HTML files.
- “Compiled” version has a .CHM extension.
- Uses HH.EXE engine and other files to run, which automatically comes with Windows 98 and NT 5.
- Can access from application, stand-alone, or from an intranet or web.
- Displays topics in a tri-pane window system, with the contents and topic windows displaying in “sync” .

Issues:

- Users must have Internet Explorer 4 or later to view.
- Can be more programmer-intensive.
- Not everyone has the “engine” to view - will have to send and include with the help file.

Strengths:

- Now established as the standard.
- Utilizes all advantages and features of HTML, as well as “help” capabilities, such as popups, etc.
- Any existing WinHelp file can be easily converted to HTML help (don’t have to “start over”).
- Can add to web site so updates to users are immediate.

Cross-Platform (WebHelp)

- Created using HTML - but uses an “uncompiled” collection of HTML files.
- Can be displayed using different Web browsers on a variety of computing platforms – including Microsoft Windows, Macintosh, and Unix.
- Cross-platform Help is viewed using the default Internet browser on the end-user’s system.

- This Help format displays in a tri-pane window similar to the Microsoft HTML Help window.

Issues:

- There is no “find” tab, so you cannot do full-text searches. However, you can use a search engine on the web site.

Strengths:

- Any existing WinHelp or HTML Help file can be easily converted (don't have to “start over”).
- Is browser-independent - will work with IE 4 and Netscape.
- Provides all standards HTML-based features and runs on Windows, Macintosh and UNIX platforms.
- Can add to web site so updates to users are immediate.
- Detects the appropriate ActiveX controls to launch IE or uses a Java Applet to launch Netscape Navigator.

Intranet Help

- Intranet content is also comprised of uncompiled HTML files placed on intranet servers for departmental or company-wide access.
- Use to create a variety of Help systems – including standalone information systems and online books.
- End users only need the ability to access the network drive or server containing the intranet content files.
- Because it's HTML-based, Intranet content is viewed using the Internet browser.

Issues:

- Users must have Internet Explorer to view.

Strengths:

- Comprised of uncompiled HTML files placed on intranet servers for departmental or company-wide access.
- Any existing WinHelp file can be automatically converted (don't have to “start over”).
- End users only need to access the network drive or server containing the intranet content files.

Windows CE Help

- Used with palmtop computers and other hand-held electronic devices.
- These are a collection of uncompiled Help files placed directly on the palmtop or electronic device.
- Contains individual .HTP files.
- “Compiled” version has an .HTC extension.

Issues:

Help generated cannot be viewed on a PC. Must copy all files to a hand held PC.

Strengths:

Any existing WinHelp file can be automatically converted (don't have to "start over").

Relax!!!

Even though there are a lot of help formats available - there are help authoring packages available that allow you to automatically generate these formats from a Single Source!!!

Considerations Prior to Creating a Help File

The following questions help determine if and what kind of Help Authoring Package is needed:

- What operating system(s) will the help file operate on (3.1, 95/NT)?
- Will you be using existing documentation, if any?
- Do you already have Microsoft Word, or another Word Processor (if needed)?
- Will you need to generate printed documentation?
- Will you need to convert Help Files to HTML?
- Will you be providing on-line tutorials?

Where Do I Start?

- Define a plan. (Audience, purpose, kind of help files, special features, help format(s), HTML, manuals, etc).
- Research and purchase a help-authoring package.
- Estimate the work effort. (How long, resources, what it will cost).
- Create the help file(s).
- Implement context-sensitive help with programmer.
- Test and distribute help file on various platforms.
- Keep up on help technology and change help as needed.

Which Help Authoring Package?

When using a lot of the Help Authoring packages, you are actually working in a Microsoft Word Document, containing text, graphics, links, etc., allowing the flexibility of creating on-line materials in the same way you create printed documentation. The document is then compiled using a Microsoft Help Compiler, and an .HLP file is created.

Note that with a help authoring package, the help compiler(s) are included. If you manually create your help files, you will need to access the Microsoft Help Compilers (HCP.EXE, HCW.EXE, HC31.EXE, HCRTF.EXE).

The following are just a few of the Help Authoring packages to quickly create help files.

Collection of help file information - www.winwriters.com

RoboHelp - www.blue-sky.com

Doc-To-Help - www.wextech.com

ForeHelp - www.ff.com

NetHelp - www.home.netscape.com

HelpBreeze - www.solutionsoft.com

Help File Components

When you open a Help system, you access the information in the Help system from **one file**. This file is the result of combining many different files together – usually working with a Help authoring tool.

Every Help system starts with ideas for communicating a message. The ideas may include text to explain concepts and tasks, images to show screens, and video and sound to illustrate a point. As the author, you work with a Help-authoring tool to create and use the elements that best suit your message.

- **Topics.** The basic unit of a Help system. Topics relay the message of the Help system, mainly through text and images. You decide the content, format, and organization of your topics.
- **Table of Contents.** When users open your Help system, they usually see a Contents tab. It looks and functions much like a table of contents in a printed book. It presents a hierarchical outline of what the Help system contains. You create the table of contents and users can browse and select topics to view from the Contents tab.
- **Index.** Users also see an Index tab when they open the Help system. The Index tab is equivalent to the index in a printed book. It displays a multilevel list of topics and keywords or phrases that you've specified to direct users to topics.
- **Full-text search.** Another tab present in the Help system is the Find or Search tab. This tab allows users to search through every word in the Help system to find topics containing that word.
- **Image and multimedia files.** By adding images and multimedia files to your Help system topics, you take advantage of the power and color of Help. Depending on the format you've chosen for your Help system, you can add graphics, sound, video, animation, and more.
- **Windows.** Windows are the frames that display topics. Each Help format has at least one default Help window in which topics automatically appear. You can customize the window's appearance and attributes. You can also design new windows to suit your content.
- **Links.** Users navigate through your Help system through links. You design how the topics are connected together. The most common links are from one topic to another. But links can also take users outside the Help system, to a topic in a different Help system (even in a different Help format), to a Web or intranet site, or to an application, for example.
- **Styles.** Topics are formatted using 'styles'. Styles are named formats that you design and apply to control the layout and appearance of text.

- **Organization.** All those idea files and other components (called source files) come together in a Help project. Your Help-authoring tool creates one file that contains the information about the location of your topics, images, and other files. Help project files also contain the settings that affect how your Help system looks and acts.
- **Help compiler.** The Help compiler isn't actually part of the final Help file, but you need it to create one. The compiler takes all the source files and other components of your project and creates one Help system file that you distribute to end-users. The compiler reads information in the project to determine what the Help system contains and how it looks and functions.
- **Help viewer.** Users work with the Help system from a Help viewer. Sometimes called a Help engine, the viewer opens and displays the Help system. You work with different viewers, depending upon the Help format.

Topics

The most basic unit of a Help system is its topics. When users view a Help system, their destination is a Help topic. Topics communicate the information or provide the assistance users are looking for. As a Help author, your job is to decide which topics meet your users needs and then organize them by "grouping" together like groups (or related sets) of topics. The grouping relationship can be based on many types of organization – frequency of tasks, related subject matter, job functions, and so on. However, the way you chunk topics affects not only topic organization, but also it often affects the way topics are linked.

Topic types

You can design the styles for your topics based upon the purpose of the topic. Here are some common types of topics:

- **Context-sensitive.** These topics allow you to provide additional assistance to users inside a software application. Users can click a Help button or select a menu command, and the application brings up a topic specific to where the user is and what the user is trying to do. There's even a special kind of context-sensitive Help called **What'sThis?** Help. It provides information about a specific item at a dialog or window such as a button option or checkbox setting.
- **Procedures.** Sometimes known as task topics, these provide step-by-step instructions for performing tasks.
- **Overviews.** Overview topics provide explanations to Help users understand new ideas and concepts. They usually begin with the familiar, and then lead the user into information they don't know, explaining terms and providing examples.
- **Definitions.** These topics usually define unfamiliar terms, but they can also define industry terms or jargon used throughout the Help system. They are often called Popup topics, because they usually appear in a small window that "pops up" when accessed.
- **Reference.** Reference topics provide non-procedural information or explanations. They often include lists of commands, shortcuts, values, parameters, or other information the user needs.

Your Help system can contain as many topics and as many topic types as you want.

Connecting topics

Help topics are connected by **hypertext links**, commonly referred to simply as **links**. Links organize topics into groups and allow users to move around the Help system and display topics. You can use many different linking strategies to creatively organize and connect your topics. One of the benefits of a Help topic is that you can "reuse" the topic over and over again without copying it, simply create a link wherever you want the topic to be accessible.

Displaying topics

Whenever a user selects a topic, from a link or from the table of contents or index, the topic displays in a window. You can add and customize the Help windows in your Help system to best suit your topic design needs.

Images and Multimedia

Images and multimedia communicate ideas – sometimes better than text. By adding pictures, sound, video, and animation to your topics, you can add interest and flair to your Help system.

Images

You can include screen shots, icons, scanned photographs, background pictures (also called watermarks), logos, and more in your Help system to give it interest and color. You can even create clickable images – images that are actually links to other topics or to Web sites on the Internet or intranet.

Keep in mind that each Help format supports different image formats, but there are many image tools available to Help you create or convert images into the format you need. (Most Help authoring tools contain image tools for that reason.)

Multimedia files

It's fun and easy to add sound, video, and animation to your Help systems. Some users aren't satisfied with just text and regular images – they want the next dimension. Multimedia files can be used to communicate concepts in a more interactive way.

Windows

Windows are the "containers" that display topics in the compiled Help system. You can control and customize the way your Help system's windows look and act:

- **Color/images.** You can add interest to your Help system by choosing background colors for your Help windows or by using images as window backgrounds (like watermarks, images that appear "beneath" the text).
- **Size and location.** You can quickly adjust the size of a window or the location in which it appears on the screen. For example, you can make Help windows overlap and position them in a specific corner.
- **Title.** Each Help window can contain a title or caption that appears in the window's title bar. You probably want the main window in your Help system to have the title describing the product or subject. For example, if you were creating an employee handbook for the ABC Company, you might title the main window something like, "ABC Employee Handbook."

- **Buttons.** Buttons on Help windows usually control the actions users can do from the window. They also control the way users move around through topics. For example, the Print button allows users to print the current topic.

Links

Links are the way users navigate throughout the topics in your Help system. Links connect topics and make them accessible. Links also group topics, so users can see the relationships between them.

Types of links

There are many different types of links that you can create, but the following are the most common:

Jumps. These are the most common and most familiar type of link. They are called jumps because when the user clicks one, the Help system "jumps" to another topic. You can use jumps to link topics within the Help system, from one Help system to another, or even to a site on the Internet or intranet.

Popups. Popups are so named because they "pop up" in a little window over the main window. Often, popups are used to define words or display notes or tips. Popups are also used to display context-sensitive Help topics from inside an application — users click on the Help button and the application displays a topic relating to the context of where they are in the application or what they're trying to do.

Hotspot images. You can create clickable images that display a topic or jump to one or more topics. These images can add interest and spark to your Help system. You can also take advantage of the truth that "a picture is worth a thousand words" by using clickable images consistent with your design. For example, rather than using the word "tip" in your content, you can include a picture that consistently represents a tip (such as a light bulb). This image provides a quick visual clue to users that the information is a tip.

Browse sequences. You add browse sequences to provide a path for viewing topics in a certain order. You group the topics in a particular order, and then you make that order available using the browse buttons in the Help window. Browse sequences provide a way for users to see and move around a group of Help topics. They are especially useful for tutorials and online training guides where users read information and perform actions in a specific order.

See Also and Related Topic links. See Also and Related Topic links provide you with another way to show the relationship between similar topics. You can use them to group topics that might be necessary or interesting to users.

Buttons. You can use existing buttons (like buttons on Help windows) or create custom buttons to help users move from one topic to the next. The most common linking buttons are the Previous and Next buttons that appear on the Help window. Using these buttons, users can move forward and backwards through a set of topics grouped into a browse sequence.

Strategies for Designing Links

After you decide the types of links to include, plan an overall navigational linking strategy. Make sure your links compliment each other and work together.

Here are some basic considerations for developing a navigational linking strategy:

- **Keep it simple and consistent.** "Less is more" is a good design rule, regardless of whether you're designing the Help interface or its navigation. Clearly identify the navigational elements, make them easy to recognize and as visual as possible. And be consistent, if you're using a button to take them to Related Topics, each time the button appears it should contain the same label and appear in the same general location. Consistency is the best way to train users how to use your system. But make sure navigation doesn't get in the way of the information. Remember, what users really want is the information. The navigational tools are simply a means to an end.
- **Provide a home base.** Give users a reliable place to begin and end. This is especially handy if users get lost. Typically, the point of entry is considered "home base", which is why the table of contents and index are so visible. Make sure users can go home from any place in the Help system.
- **Avoid over-navigating.** Although it's important to provide a variety of pathways, do not offer too many choices to avoid overwhelming users. Make sure your navigational aids aren't just repeating topics in an effort to appear comprehensive.

Table of Contents

Users are familiar with a table of contents because they are standard in printed materials and in most online documents. Tables of Contents (TOCs) allow them to get an idea of the overall outline and organization of the Help system. Users can browse through the TOC to find a topic or to see the relationship between topics.

The Help system's table of contents is contained in a file called the Contents file (.CNT). This file controls the appearance and attributes of the Contents tab — the tab that displays the Help system's contents. The Contents file uses the books and pages metaphor:

- **Books.** Group topics into chapters or steps. Users click on a book to display its contents, which are represented by pages.
- **Pages.** Represent individual topics. Users click on pages to display topics.

You can arrange the books and pages in any order. You can include all the topics in your Help system or only those topics you think users are most likely to need.

Index

Users often search through the index of a printed book to find the page containing the information they're looking for. In a similar way, your index allows users to search for information too, but much more interactively. In fact, studies show that the index is used more frequently to find information than the table of contents or full-text search. An index allows users to quickly get to the information they need and want.

The Help index displays a multilevel list of topics and keywords or phrases that you create. There are two ways users get to topics using the index:

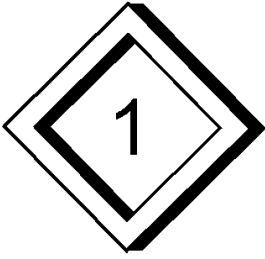
- **Typing.** Users can type a keyword or phrase and go directly to a topic or to a list of topics containing that keyword or phrase.
- **Browsing.** Users can also browse through the index, and then select a keyword or phrase. They either go directly to the topic or to a list of topics containing the selected keyword or phrase.

You can have as many keywords and phrases as you want or need. When creating an index, don't just limit yourself to terms inside the Help system. It's a good idea to cross-reference keywords with synonyms. Carefully consider the words or phrases according to your users' ways of thinking.

Macros

You can extend the functionality of your WinHelp Help system using macros. Macros are coded scripts that perform certain functions, like providing access from a topic to a Web site on the Internet. Both WinHelp 3 and WinHelp 4 Help systems support macros, although WinHelp 4 systems support a wider range of WinHelp macros.

This Page Intentionally Left Blank



Step 1: Creating a Help File

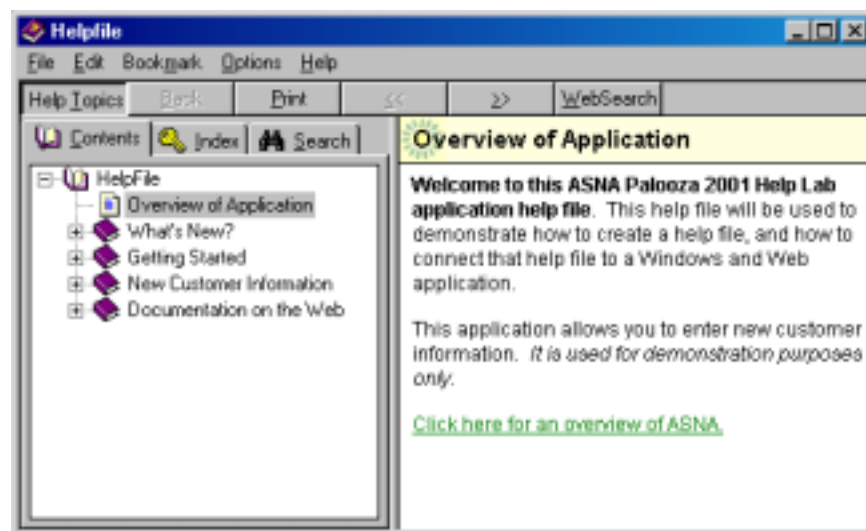
What you will learn in Step 1:

- How to create a new help project.
- How to create help topics, and add jumps, popups and buttons.
- How to create WhatsThisHelp topics.
- How to create a Contents Tab.
- How to insert Multimedia files.
- How to change Project Settings.
- How to compile a Help Project.
- How to create and edit a Map file.
- How to generate WebHelp.

Approximate Time to Complete Step 1:

Approximately 1-1/2 hours.

What the Help File will look like after completing Step 1:



What We Will Cover in Step 1

In this step, we are going to create a Help file to be used in conjunction with an AVR Application (see Step 2) and a Web Application.

The Help file will contain topics, jumps, popups, HTML topics, graphics, and macros.

The Help file will be accessed from a Help Menu, (accessing the entire help file), as well as utilizing context-sensitive help from a Help button and from each IOField by pressing F1 or the WhatsThisHelp button.

The application we will be creating the help file for is shown below.

The screenshot shows a Windows application window titled "Connecting Help to an Application". The window has a menu bar with "File", "Edit", "View", "Tools", "Window", and "Help". The main area contains a form for "New Customer Information". The form has a title bar with "ASNA" on the left and a graphic of a person on the right. The form fields include "Cust No.", "Name/Address" (with sub-fields for Name, Address 1, Address 2, City, State, and Zip), and buttons for "OK", "Cancel", and "Help".


The Help file will need to have the appropriate pieces in place in order to do the following:

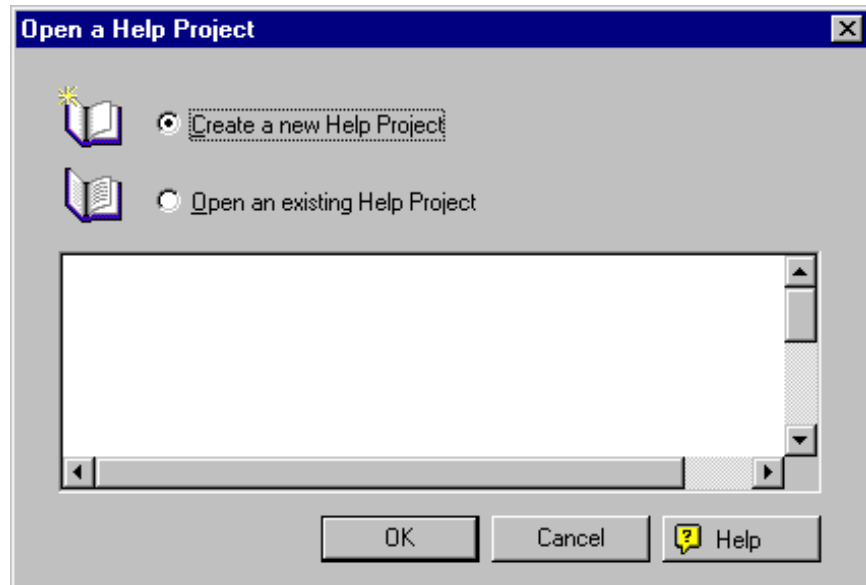
1. Call the entire Help file from the **Help** menu.
2. Call a topic called **New Customer Information** when the **Help** button is pressed.
3. Display individual help topics using the **HelpContextID** and **HelpKey** properties, as well as display **WhatsThisHelp** popups using the **WhatsThisHelp** ID property.

Creating a New Help Project

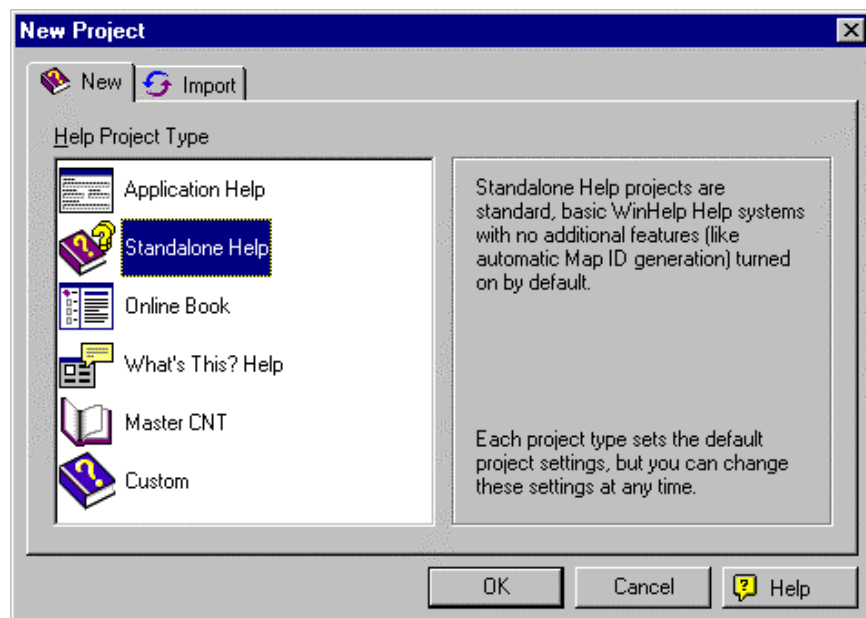
To create a help file, you actually create a **Help Project** that contains all of the topics and files needed to create a help file. The Help Project's information is stored in the **.HPJ** file. There are various files created that all make up a help project, such as .RTF, .HH and .HPR.

To Create a New Help Project

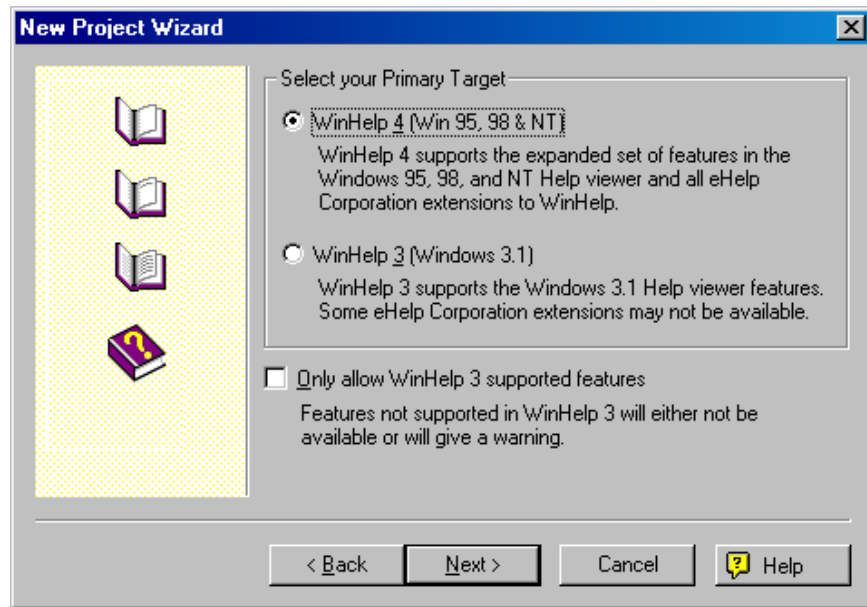
1. Create a RoboHelp Project by selecting the **RoboHELP** icon . The following dialog will display.



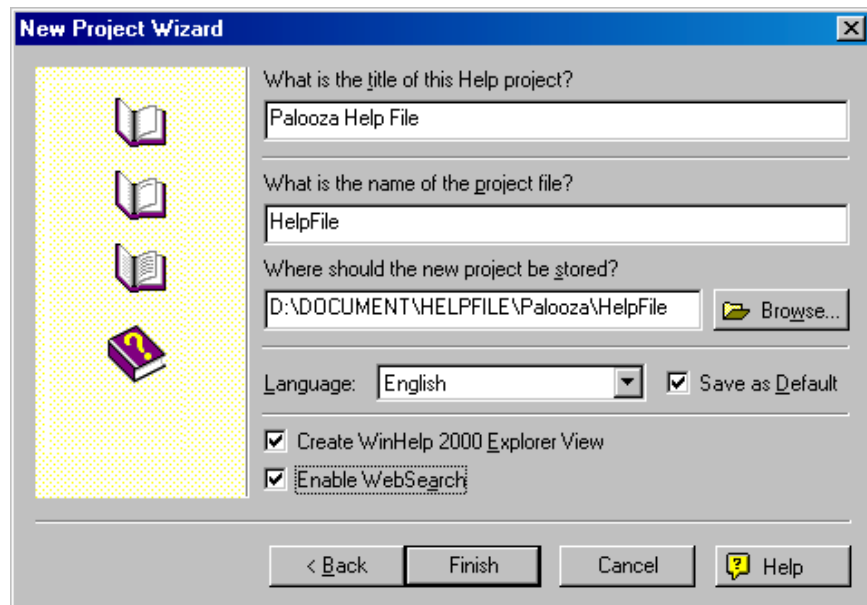
2. Select **Create a new Help Project** and select **OK**. The following will display.



3. Select **StandAlone Help** and select the **OK** button. The following will display to set your type of help file.

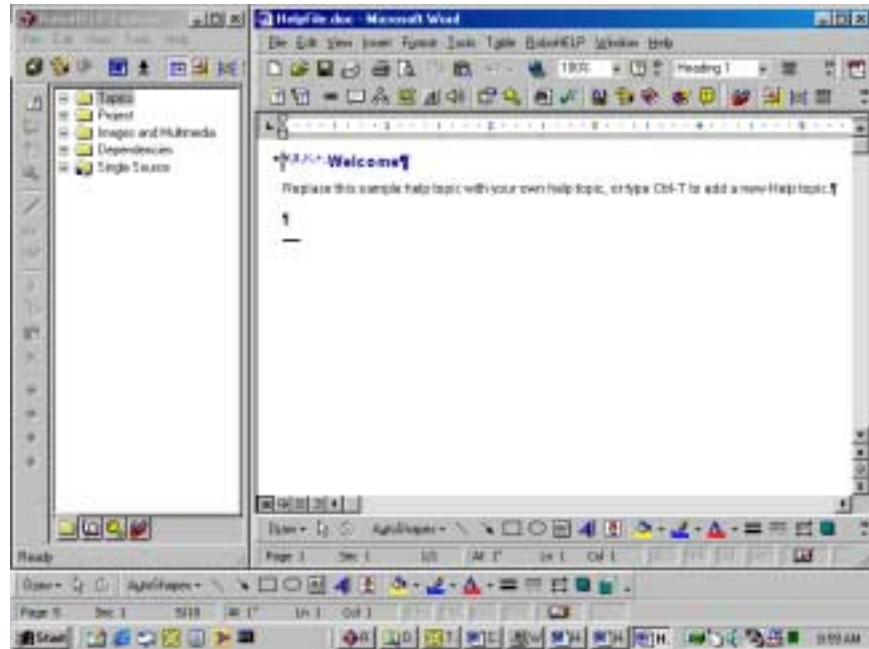


4. Ensure that **WinHelp 4** is selected as your Primary Target, and select the **Next>** button.
5. A dialog will display for you to enter a Title and File name for the help file. Enter the information as shown below.



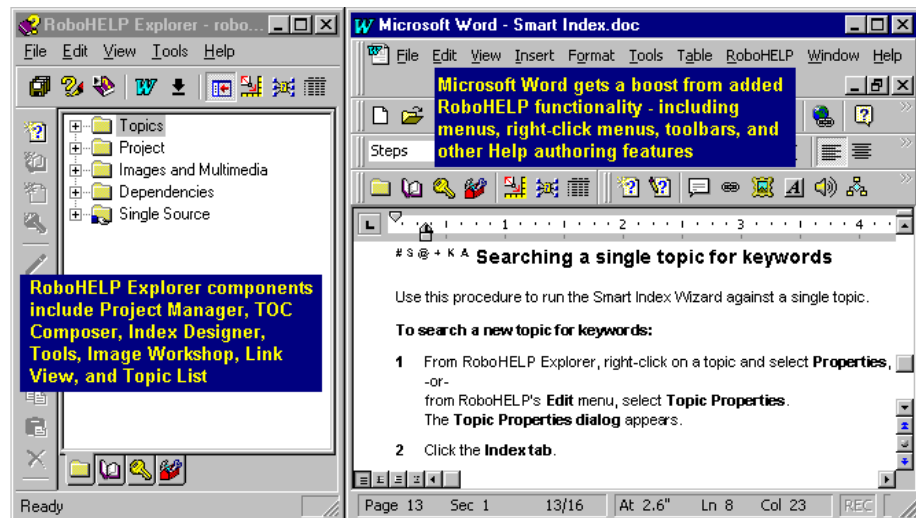
Title of Help project?	Palooza Help File
Name of Project file?	HelpFile
Location to Store Project	Palooza/HelpFile
Enable WebSearch	Select Enable WebSearch

6. Select the **Finish** button when complete. RoboHelp will display, displaying the following Windows.



Getting Around in RoboHelp

When you open RoboHELP, you see RoboHELP Explorer and RoboHELP with all their working components. After you open a WinHelp project or create a new project, you can select these components and work with them.



RoboHelp Explorer (Left Side)

RoboHELP Explorer allows you to manage and organize all the elements and source files of your Help projects. It uses the familiar **Windows Explorer** metaphor to make it easy for you to see the overall project view and can quickly make the necessary decisions about all the elements of your Help system. You can resolve broken links, manage Map Ids, define windows and colors, design a Table of Contents, Index, print reports, and much more.





Menu Options

Click on the Menu options for an idea of the options available.

Toolbar on Top and Side

Quickly select the icons on the top and side. The icons on the side will vary depending upon the icon component selected on the bottom.

Icons at the bottom

1. Click on the **Project** tab , then select the Project folder, displaying folders for sections that make up the project file.
2. Click on the **TOC** tab , which will contain the “Contents” tab editor in which books and pages are created to display when the help file is opened. A separate file is created containing a **.CNT** extension.
3. Click on the **Index** tab , which contains the keywords that will be listed in the Index tab. (These keywords are used with the **HelpKey** property in AVR).
4. Click on the **Tools** tab  that displays some advanced tools that RoboHelp has available, such as a Knowledge base and Video Capturing software.

Microsoft Word (Right Side)

RoboHELP is fully integrated with **Microsoft Word**. This allows you to work in a familiar Word environment while developing your Help system's content. You create topics, insert links, add images and multimedia, insert buttons, and more with RoboHELP inside Microsoft Word.

Menu Options

Click on the Menu options to see the standard Word menu options, along with the RoboHelp menu options at the end of each menu option.

Toolbar on Top

RoboHelp-specific icons have been added directly below the menu options for quick accessibility.

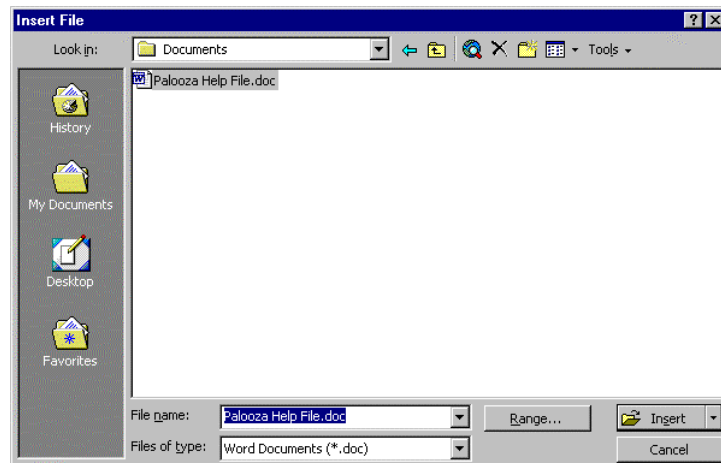
Inserting an Existing File

Existing text can be inserted into a help project and automatically be converted to a help file. Any file format that Word imports can be used, such as another DOC file, HTML file, RTF file, and text files.

Tip: Even if you don't currently have RoboHelp or another help authoring package, you can still begin developing documentation, even if you enter it into notepad, or another file format that Word will import.

To Insert an Existing File

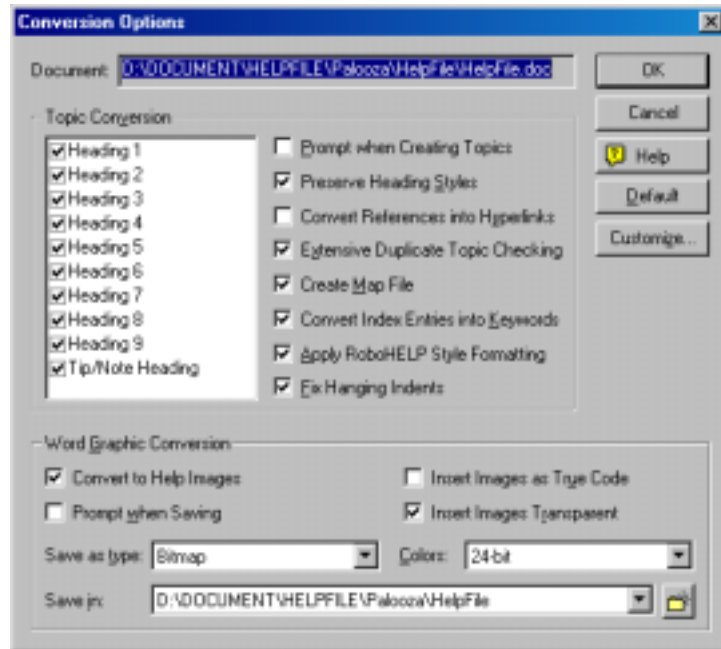
1. Highlight the entire **Welcome** topic and press the **Delete** key.
2. Go to the Word document (on the right side) and select **Insert – File - \Session 05 – Creating Help Files\Examples\Help_File\Source Files\Palooza Help File.Doc**, then select the **Insert** button.



A word document will be inserted, containing text and pictures.

Each topic, or title was previously assigned a “**style**” called **Heading 1**. A style is merely a ‘name’ given to a particular set of font attributes, such as point size, font type, style, paragraph spacing, color, tabs, etc that you want to repeat over and over. Applying a ‘style’ saves you time in manually changing text to the same attributes over and over again. Word has several built-in styles that can be used. Heading 1 is **Ctrl+Alt+1**, Heading 2 is **Ctrl+Alt+2**, etc.

3. Select **RoboHelp – Convert Document**. The following dialog will display.



4. Keep the defaults, and ensure that **Create Map File** is selected and select **OK**.

RoboHelp will work for a few minutes. It will convert the images to graphics, and change each topic noted as a Heading style to a **Help Topic**, which is noted with Footnotes. These footnotes tell the compiler what properties that help topic has, such as what type of topic it is, it's keywords, browse sequences, etc.

See the **next page** for more information.

Saving the File



1. Save the file to add the new topics to the file by going to RoboHelp Explorer (left side) and selecting **File – Save All**, or **Ctrl+S**.

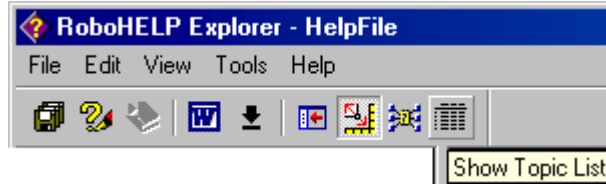
Creating an Index

When an imported file gets **converted**, keywords for that topic are not automatically generated as they are automatically when you 'create' a new topic. So the following steps allow you to quickly create keywords for the existing topics.

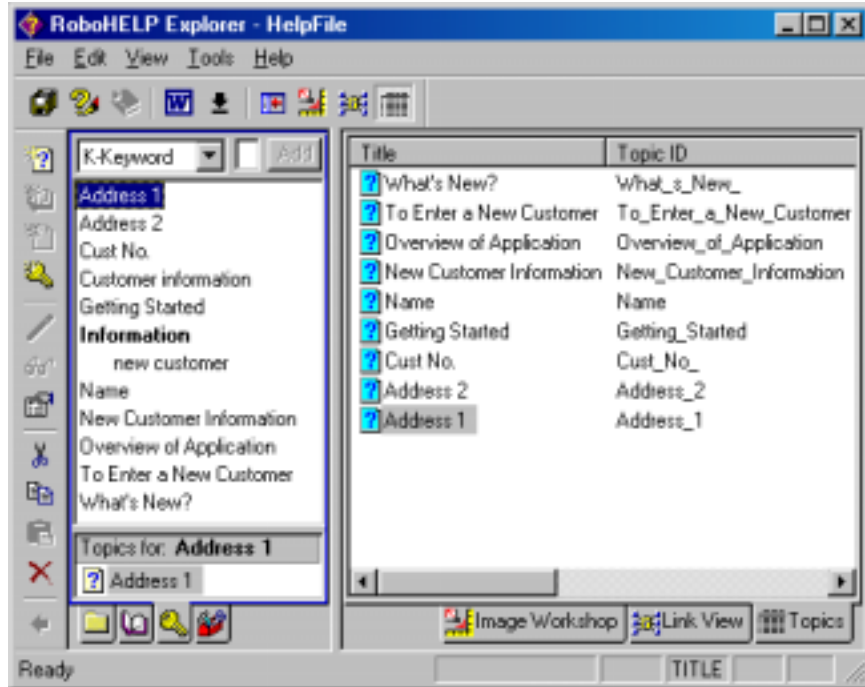
After generating an index for a topic, the “**K**” footnote will display next to the help topic. The Index tab contains a listing of Keywords that were defined for each help topic. Users use these keywords to search on a particular topic.

To Automatically Create an Index for Existing Topics

1. Select the **Index Tab** with the Key icon .
2. Select the **Show Topic List** icon .



3. Select **All** of the items on the right and drag-and-drop onto the window on the left.




The default keywords (same as the topic title) will display in the Index tab. The icons to the left of the topics in the Topic List will change to light blue, indicating that those help topics have a corresponding keyword in the Index tab.

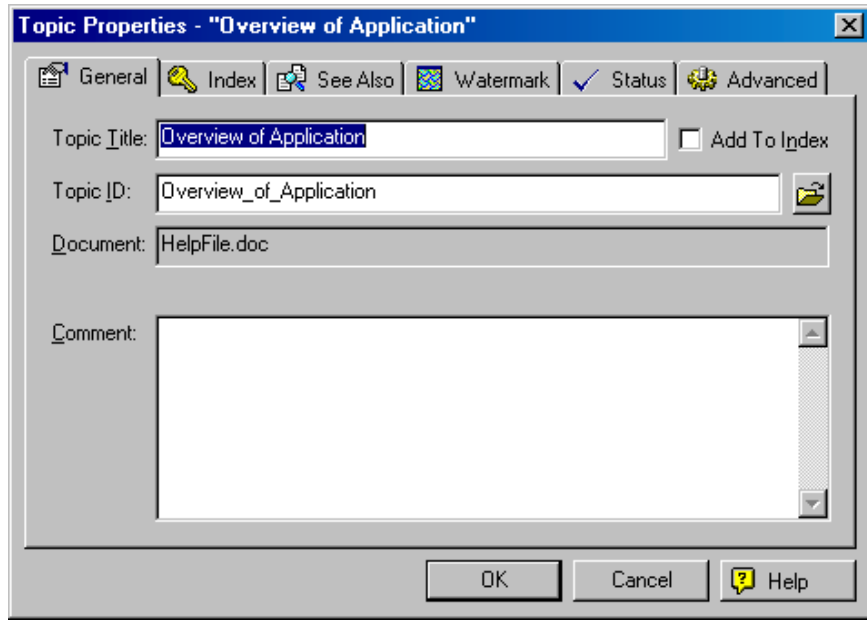
4. Select the **Hide Right Pane** or **Show Word** icon to display the help project again.

Now you will see a 'K' footnote added to the footnotes to the left of each help topic.

Properties of a Help Topic

To View the Properties of a Help Topic

1. Put your cursor on the first topic (**Overview of Application**) and select the **Topic Properties** icon  (second from the left). The **General** tab information will display.



The following is a listing of the various footnotes you may see, based upon the settings for a topic.

#	Topic ID (Required footnote)	<p>The Topic ID is the internal Name given for that help topic. It uniquely identifies a particular topic among all the topics in the Help system. You use the topic ID when you create a link or topic reference to refer to the topic to display.</p> <p><i>Note: In AVR, when using a Help button on a form, you can always refer to or call a particular help topic by using the Winhlp32 parameter -I, followed by the topic ID. Also note that the topic ID contains underscores for spaces, and the topic ID gets stored into the RTF file.</i></p>
\$	Text on Screen (optional)	<p>This is the title that the user sees on the screen when a topic is selected.</p> <p>It is also used to list a topic in the WinHelp viewer when users perform a search.</p> <p>Titles are optional for topics, as you may not want to use a title for popup topics.</p>
K	Index Keywords (optional)	<p>Identifies words or phrases that users can enter to search for a particular topic in the Help system's index (on the Help system's Index tab).</p> <p>Index keywords are optional. You may not want to use them for popup topics, and are not available for WhatsThisHelp Popup topics.</p> <p><i>Note: In AVR, a Keyword string is used to jump to a particular topic using the HelpKey property.</i></p>

+	browse sequence (optional)	A number assigned to the topic to specify a suggested reading order, so users can browse through related topic content. Users access browse sequences through Browse buttons (Next >> and Prev <<) from the window's button bar. Browse strings are optional, and often omitted for topics that you only want to display in popup windows.
@	Status of Topic (optional)	Allows you to keep track of the status of each topic. Project Management Reports can be printed out.
>	Default window (Secondary)	Specifies the default topic window used to display topics accessed from the Index, Find, or Search tabs in the Help system.
!	Topic entry macro	Execute a macro as soon as the topic is displayed (selected by users). Topic entry macros are similar to program scripts or batch files and can be useful when you want to customize a specific topic.

Adding New Topics

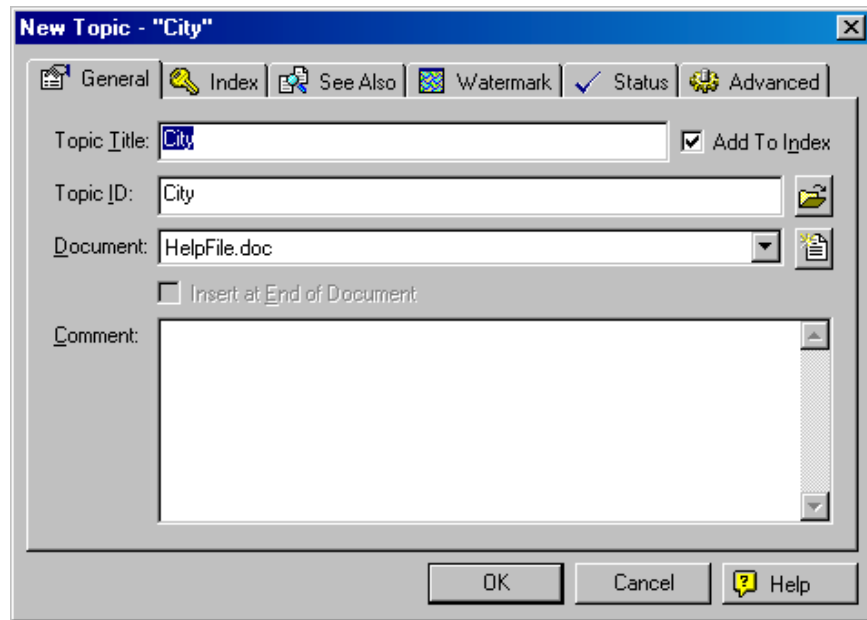
For our AVR and Web applications, we need context sensitive topics for **Cust No., Name, Address 1, Address 2, City, State and Zip.**

Note that the only topics we will need to add for this step is **City, State and Zip**, as the others have already been added.

We will also add each of them as a **WhatsThisHelp** topic later in this step.

To Add a New Help Topic

1. Go to end of document. Select the **New Topic** icon , or select **Ctrl+T**. The New Topic dialog will display.





2. Enter in the following **Topics** and their text descriptions and select **OK**.


Topic Title	Description
City	Enter the name of the city in which the customer resides, entering up to 30 characters.
State	Enter the 2-character representation for the state in all caps.
Zip	Enter the complete 10-digit zip code, being sure to enter the hyphen, such as 78216-0001.

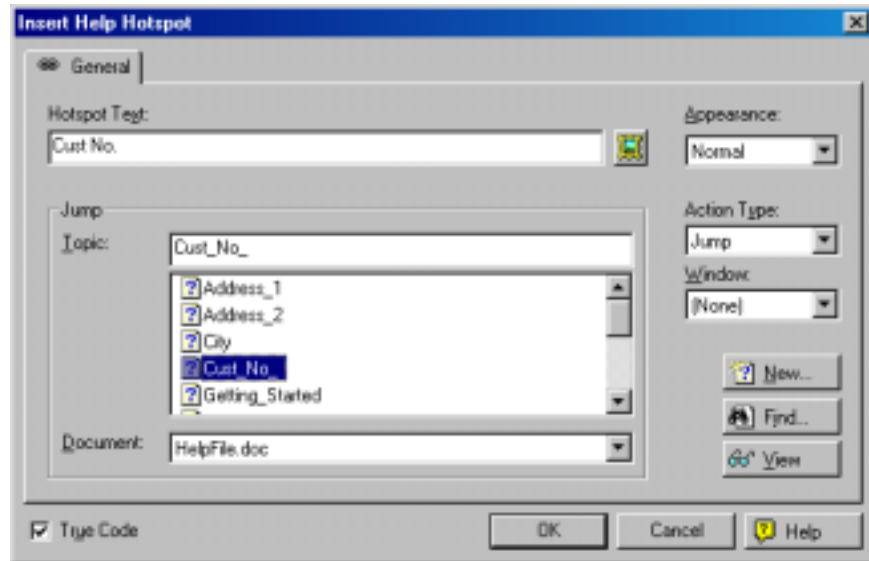
Note that when you enter the Topic title, it also defaults to the Topic ID. You may want to add all the topics first, then go back and add the description for each.

Creating Links

Now that you have topics, you can create jumps or popups to those topics using the Jump  or the Popup icon .





To Create a New Link

1. Go to the **New Customer Information** help topic, and make jumps by performing the steps below.
2. Highlight **Cust No.**
3. Select the **Jump** icon  or select **Ctrl+J**.
4. In the **Jump Topic** field, begin typing in **Cust** until a match is found to jump to the help topic **Cust_No_** and select **OK**.



Note that Cust No. is now in green text with a double-line, indicating it is a jump. The help topic of **Cust_No_** will display in black immediately after the first Cust No, indicating that when the jump Cust No. is selected, the user will be taken to the Topic ID called Cust_No_, which is correct.

5. Perform the same steps for the other topics as listed below:

Topic	Type of Link	Jump Topic ID
Name	 Jump	Name
Address 1	 Popup	Address_1
Address 2	 Button - Authorable	Address_2
City	 Button - Mini	City

6. Highlight the **Address 1** help topic, and select **Format – Paragraph – Line and Page Breaks** tab and deselect the **Keep with Next** field.

Creating Context-Sensitive Help

Now we have entered all the Help Topics for our application. A user can search on all fields, even if the application is not open. We can use the **HelpContextID** and **HelpKey** properties to display the associated help topic. Note that these properties actually launch the Winhlp32 engine, the entire help file is launched, and the appropriate topic is displayed.

Again, creating a standard Help Topic generates a “jump” to that topic using the above properties.

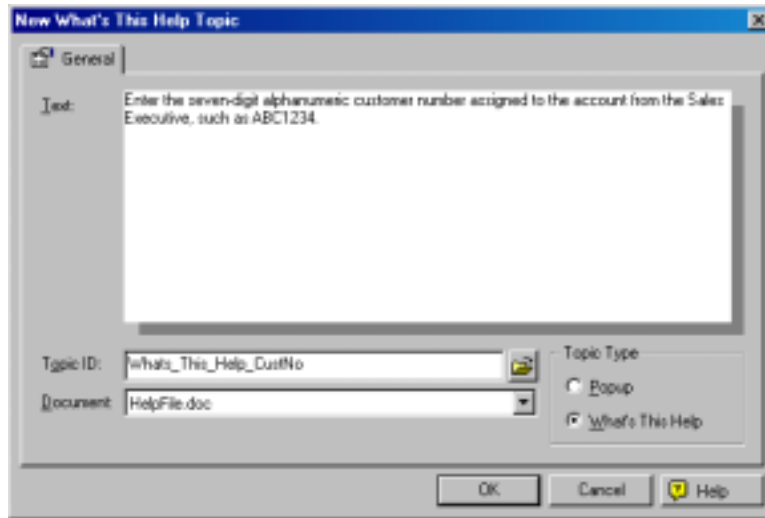
If you want information for a help topic to display in a 'popup' window, you can create what is called as "**WhatsThisHelp**". WhatsThisHelp topics provide brief explanations for the selected field in a popup window. **Winhlp32 is not launched**, so the help file is not displayed. Only the specified help topic is displayed in a popup window, and that window closes when any other button is selected.

 **To Create 'WhatsThisHelp' Popups**

In this step, we will create **What's This Help Popups** for **Cust No., Name, Address 1, Address 2, City, State and Zip**.

1. Highlight all seven topics listed above and **copy** them to the end of the document.
2. Highlight the text for **Cust No** and do an **Edit – Cut**, or **Ctrl+X**.
3. Select **Insert – Special Help Topics – Whats This Help Topic (Ctrl+W)**
4. **Paste** in the **text** and enter a topic ID of **Whats_This_Help_CustNo**, as shown below, and select **OK**.

Note: You do not have to put in the underscores in the Topic ID. They will automatically be there when the space bar is pressed.



5. Repeat steps 2-4 for the rest of the topics listed below.

Help Topic	Text	Topic ID
Name	Enter the name of the company here. This field is 30 characters long.	Whats_This_Help_Name
Address 1	This is the first line address of the company, entering up to 30 characters. It is used on any reports that print your company address.	Whats_This_Help_Address1

Address 2	This is the second line of the address of the company, entering up to 30 characters . It is used on any reports that print your company address.	Whats_This_Help_Address2
City	Enter the name of the city in which the customer resides, entering up to 30 characters.	Whats_This_Help_City
State	Enter the 2-character representation for the state in all caps.	Whats_This_Help_State
Zip	Enter the complete 10-digit zip code, being sure to enter the hyphen, such as 78216-0001.	Whats_This_Help_Zip

6. Remove all extra titles and new line markers and change the **style** to **normal**.

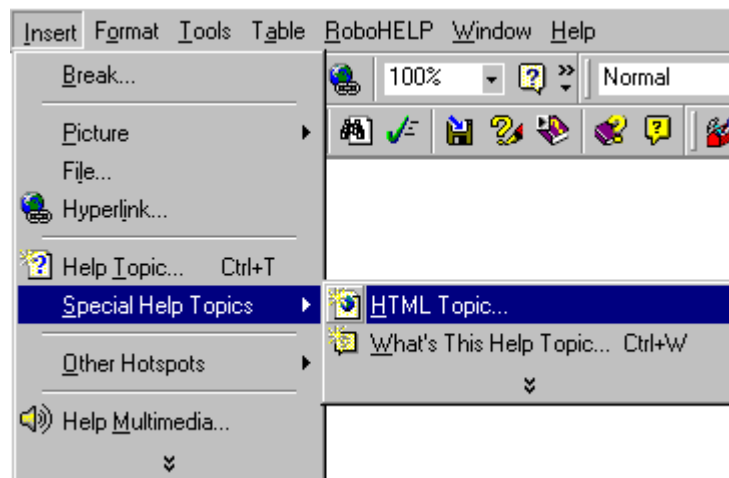
Adding HTML Topics

You can utilize or link to an **HTML** file to give your users access to local HTML files or any Web pages. Once you've inserted an HTML topic, it becomes almost like any other topic. You can include it in the table of contents, index it, create hotspots to its WinHelp Topic ID, and even include it in a browse sequence.

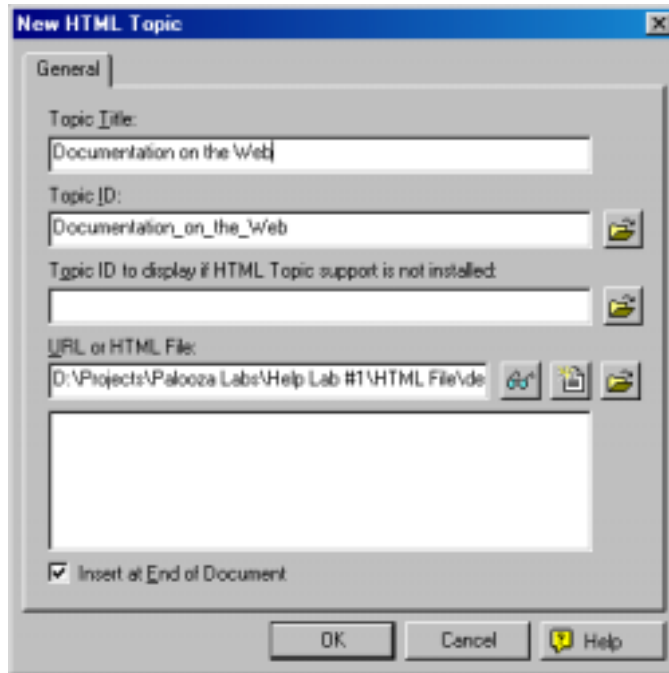
This topic type makes use of the **JumpHtml WinHelp** macro, which requires Internet Explorer 4 or higher to run and HTML support (RoboEx32.DLL). However, RoboHELP allows you to choose an alternate WinHelp topic to display in case not all your end-users have these requirements. You can also use alternate HTML jump options to display the HTML topic in the users' default browser.

To Add an HTML Topic

1. Go to the end of the help file and select **Insert – Special Help Topics – HTML Topic...**



2. Enter the following in the dialog, as shown below:

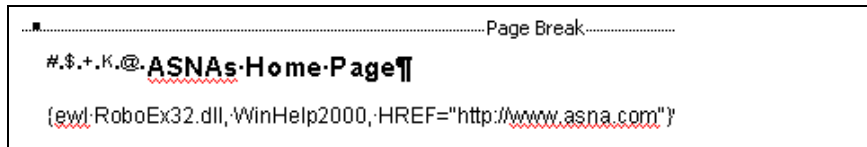


Topic Title:	Documentation on the Web
Topic ID:	Documentation_on_the_Web
URL or HTML File:	Session 05 – Creating Help Files/Examples/Help_File/Source Files/default.htm <i>Could also enter a URL, such as: http://www.asna.com</i>

3. You should now see the following in your help file:




Note that if you entered a URL instead, the HREF would show a hyperlink reference to HTTP, as shown in the example below.



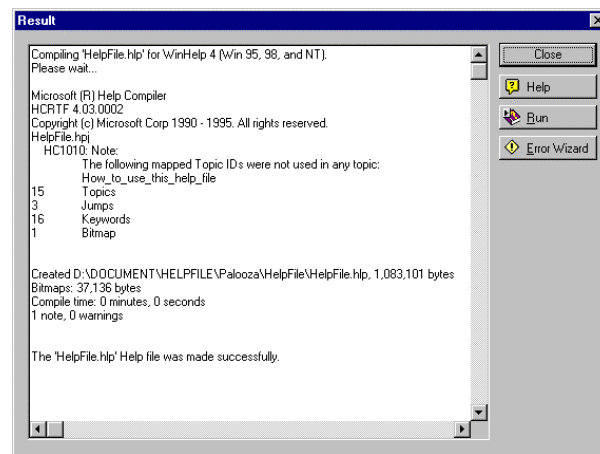
Compiling the Help Project


Compiling your Help project converts the Word document into a Help file. The selected WinHelp compiler (based on the currently selected Primary Target), takes **all** your project source files – topics, links, images, multimedia, buttons, windows, and so on, and creates a compiled WinHelp Help (HLP) file. When you select the Compile command, RoboHELP automatically saves any unsaved changes to your Help project before the WinHelp compiler begins.

To Compile a Help Project

1. Select **File – Compile** from RoboHelp Explorer, select **Ctrl+M**, or select the Compile icon  in Microsoft Word.

The help file will compile, and a Window similar to the following will display.



2. Select the **Run** button  to view the Help File. Notice that the Index tab displays on the left side. You will notice that there is not a Contents Tab, as we will create one on the next page.




Creating a “Contents” Tab

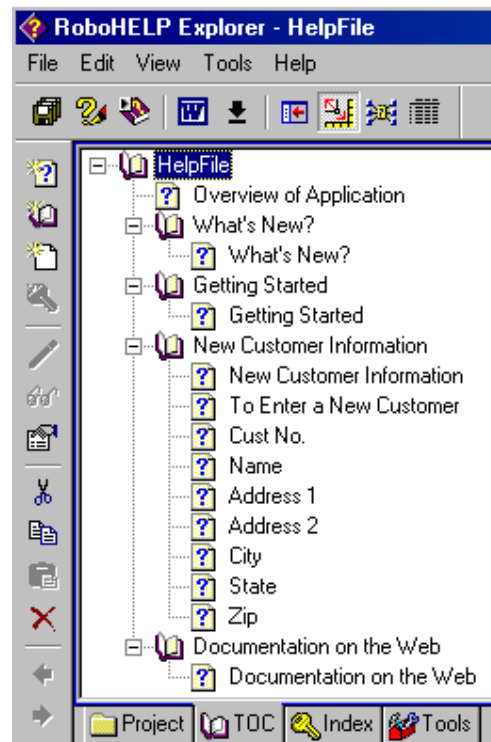
In this section, we will create the “**Contents**” tab for our help file. This is done by using RoboHELP Explorer’s TOC editor. This editor allows you to add books and pages that link to a topic, and rearrange the order as needed. You can also include a Contents tab file (.CNT) included with another help file.

In this step, we will automatically generate a TOC based upon our existing help topics in the help file.

To Create a Table of Contents

1. Select the **Table of Contents** book  in RoboHelp Explorer.
2. Select **Tools - Auto Create TOC**.
3. Double-click on the **HelpFile** book.
All of the help topics will display as a ‘page’.
4. Select **What’s New?**, then select **Edit – Create book from Page**.
5. Select the page **Getting Started**, then select **Edit – Create Book from Page**.
Then select the left arrow to move the book over.
6. Select **New Customer Information**, select **Edit - Create Book from Page**.
Then select the left arrow to move the book over.
7. Select **Documentation on the Web** and select **Edit - Create Book from Page**.
Then select the left arrow to move the book over.

The Table of Contents should now appear like the following.



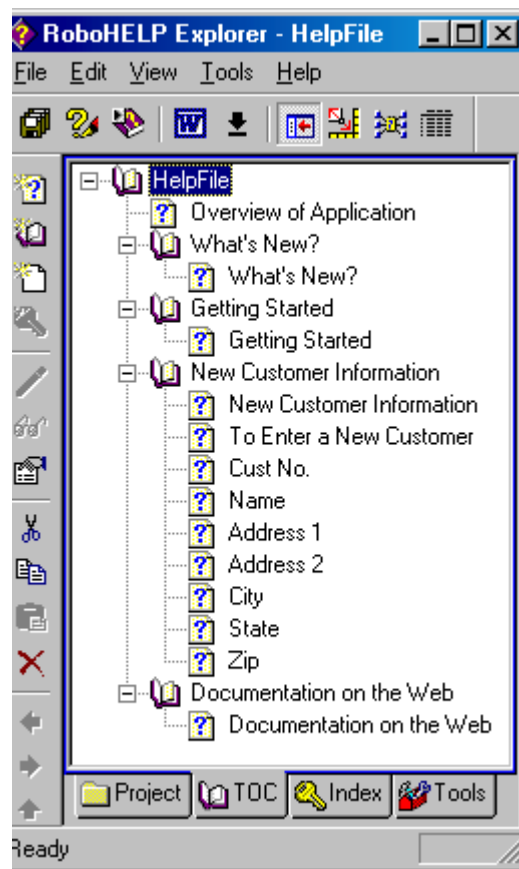
Adding a Graphic

Graphics, or images can very easily be placed into a help file. You can capture an image by selecting the **Alt+Print Scrn** button, and then paste the image into The **Image Workshop**.

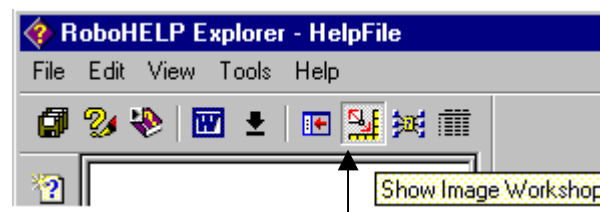
In the **Image Workshop**, you can open, edit, save, crop, and resize images.

To Add a Graphic

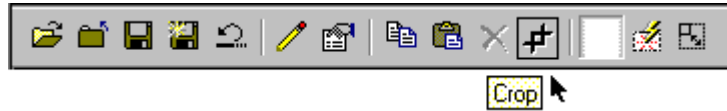
1. Select the **New Topic** icon or select **Ctrl+T** and enter a topic title “**A Graphic**”
2. To add a graphic, click on the **RoboHELP Explorer** window to make it the active Window.
3. Select the **Alt+Print Scrn** button. The image will be copied to the Clipboard.



4. Go to the end of the Word Document and create a new topic called **Graphic**.
5. **Paste** the graphic by selecting **Ctrl+V**.
6. To edit, crop or save an image, use the **Image Workshop**. Open the Image Workshop by selecting the Show Image Workshop icon, as shown below.



7. **Paste** the copied image into the Image Workshop by selecting **Ctrl+V**.
8. Select the **Crop** tool, as shown below.



9. “Draw” around the icon(s) or graphics you could like to capture.



10. Select the **copy** icon, or select **Ctrl+C**.
11. Select the **Close** icon, and then **Paste** the graphic again.
12. Select the **Save** icon and give the graphic a name.

You can also edit, resize or open an existing image.

Inserting Multimedia Files

Adding multimedia can add usability and sophistication to a help file. Used wisely, multimedia is an effective way to add demonstration and communicate to your users. You can easily add sound and video to your Help system.

WinHelp supports the following **video** and **sound** files:


AVI files are the standard video and animation format for Microsoft Windows and WinHelp. These video files offer full motion video and audio without special hardware. You can create AVI files by converting a video camera tape using a video capture card or by using RoboHELP Software Video Kit.

WAV files are the native sound format for Microsoft Windows. These audio files are digitally sampled sound clips that work well when you need realistic sound effects, spoken voice, and music. Waveform audio files tend to be high quality but are usually large files, even when compressed. You can capture WAV files using a microphone or other device with utilities (like the Microsoft Windows SNDREC32.EXE) provided with your computer's sound card.

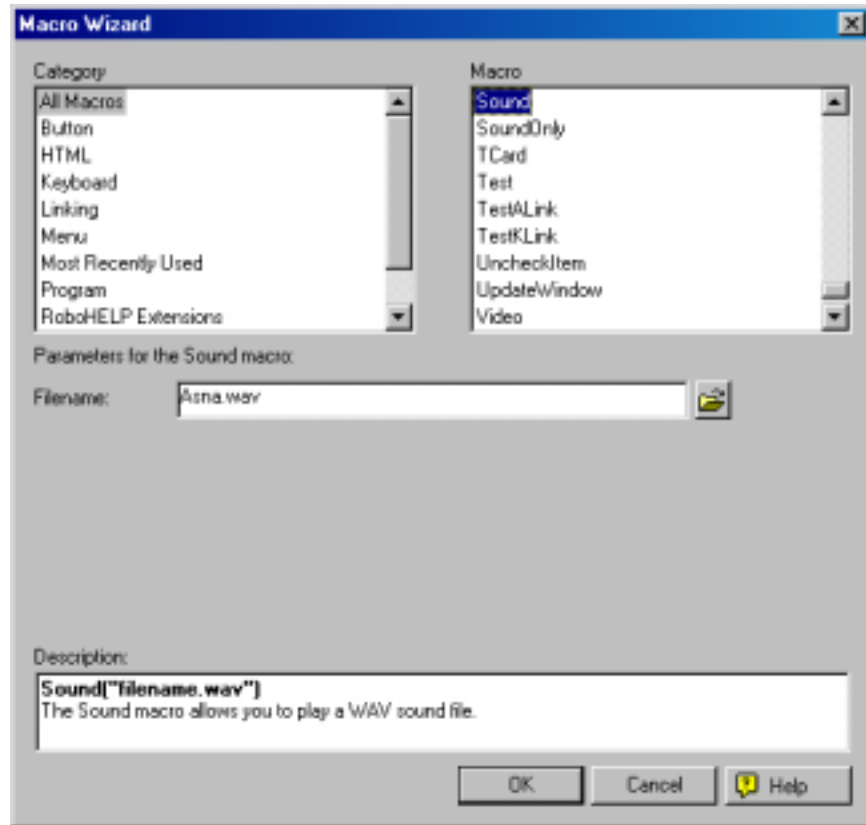
To play a sound file, users must have a sound card with the appropriate software drivers installed on their computer. They also need to have some sort of speaker system, internal or external, to hear the sound.

Inserting a Wave File

To Insert a Wave File

1. Go to the first topic **Overview of Application**.
2. Highlight the last sentence “**Click here for an overview of ASNA**” .
3. Select the **New Macro Hotspot** icon .
4. Select the **Wizard** button on the left.
5. Scroll down on the right and select the **Sound Macro**.

6. Select the **Browse** folder icon and select the **Asna.wav** file (located in the **\Session 05 – Creating Help Files\Examples\Help_File\Source Files** folder, as shown below and select **OK**.



7. Select **OK** again. You will now see the following on your screen.

[Click here for an overview of ASNA!Sound\("Asna.wav"\)](#)

Window Settings

In the **WINDOWS** section of the Project file, you can define the size, location, and colors for the **Main** Help window and any **secondary** windows used in a Help file.

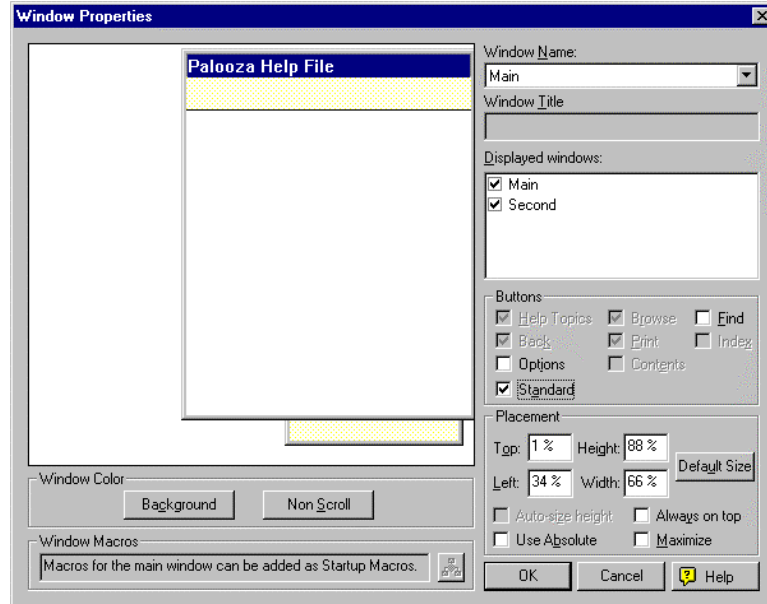
While you can only have one **Main** window per Help project, you can have several secondary windows, in fact, you can define up to 255 secondary windows.

Secondary windows look virtually identical to the **Main** window, except they don't include the menu bar. Options for secondary windows differ based upon the currently set Primary Target. You can change the window title, adjust the size and location of a secondary window, select or change the title for a secondary window, and select the colors of the non-scrolling and background regions.

In this section, we will change the default **Main** Window size and non-scrolling region color.

To Set Project Windows Settings

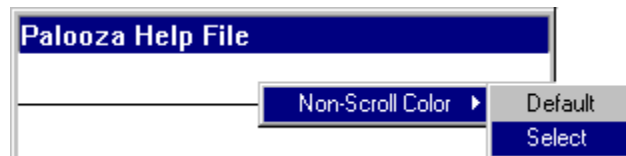
1. Select the **Project Settings** icon in RoboHelp Explorer.
2. Select the **Project – Windows – Main** folder.



3. Change the Window size by either stretching and moving the Window to the desired size and location, or entering a value within **Placement**. Key in the following numbers:

Top	1%
Left	34%
Height	88%
Width	66%

4. Select the **Non-Scrolling region** and click the **right-mouse button**. Highlight Non-Scroll Color, select **Select**, then select the light yellow color and select **OK**.



Notice that there is always a secondary window available (Second) by default that you can use to display your help topics. We will use the Secondary window in our AVR application in the next step.

5. Select **OK** to close the Window dialog.

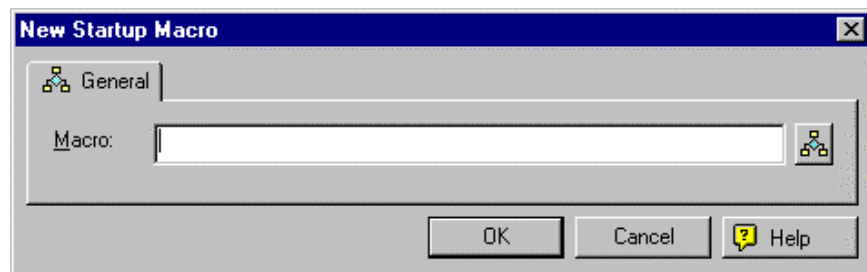
Changing the Background Color of a Popup

Use this procedure to add a startup macro in which the **background** color of a popup is light yellow. This color is a standard, however, any color could be used, if desired so the background of the popup is different than the background of the application.

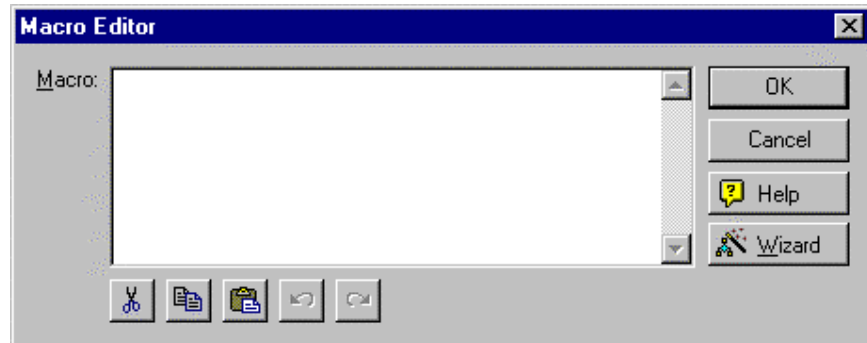
A **Startup macro** is available to the entire Help file as soon as the Help file is opened.

To Create a New Startup Macro

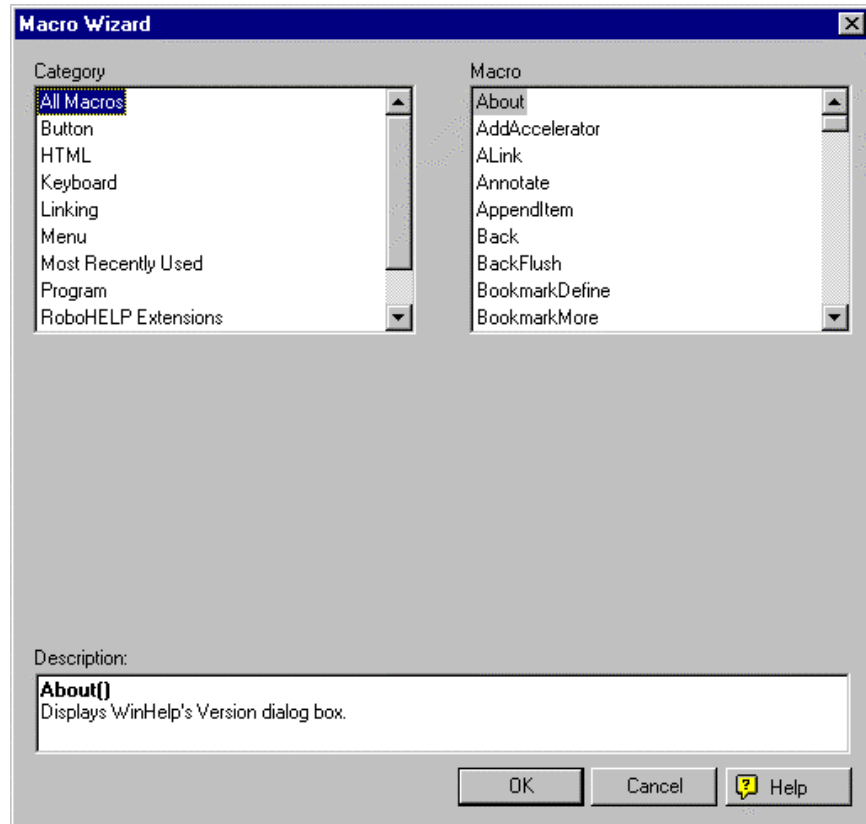
1. Select the **Project** icon in RoboHelp Explorer.
2. Select the **Project Folder**.
3. Select the **Startup Macros** folder, click on the **right-mouse button** and select **New Startup Macro**.



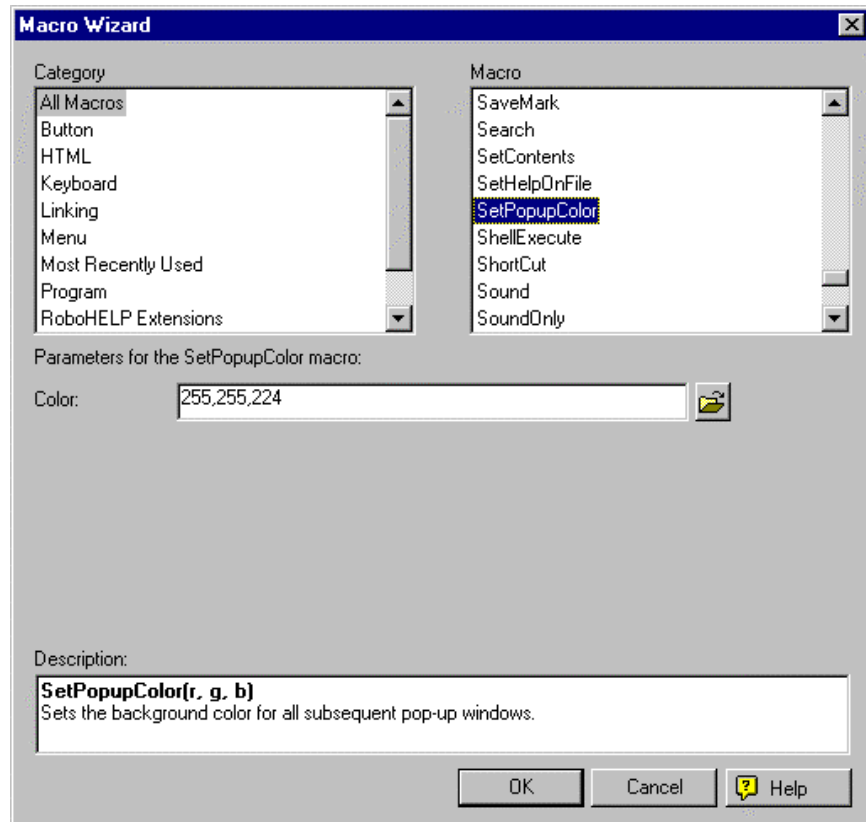
4. Select the **Macro** icon at the right of the dialog. The following will display.



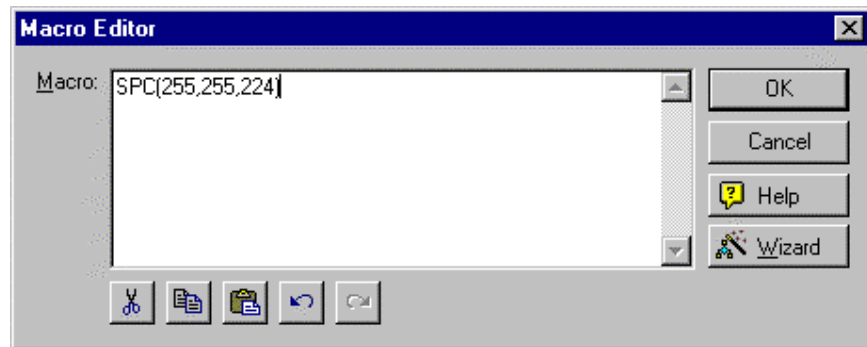
5. Select the **Wizard** button at the bottom right. The following will display.



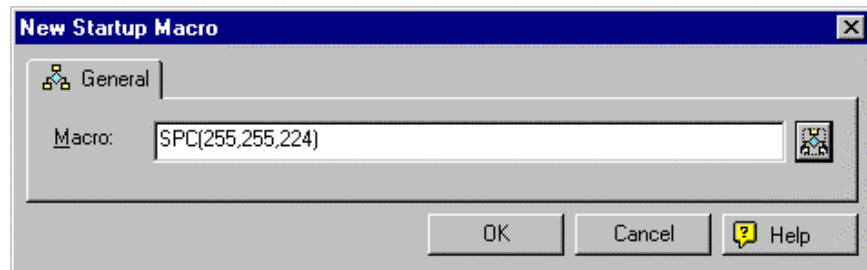
6. Scroll down in the **Macro** window on the right and select **SetPopupColor**.



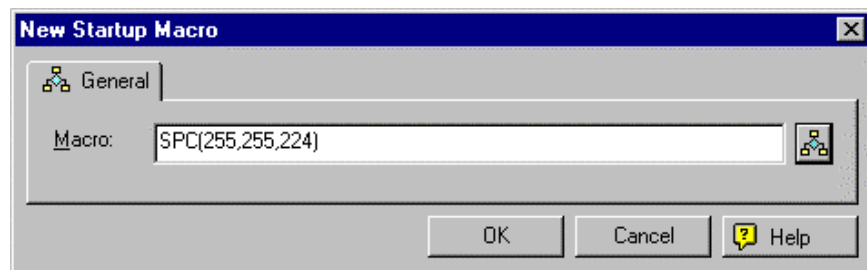
- Enter **255,255,224** as the R,G,B colors and select **OK**.



- Select **OK**.



- Select **OK**. The new startup macro lists at the end of the Startup Macros folder.

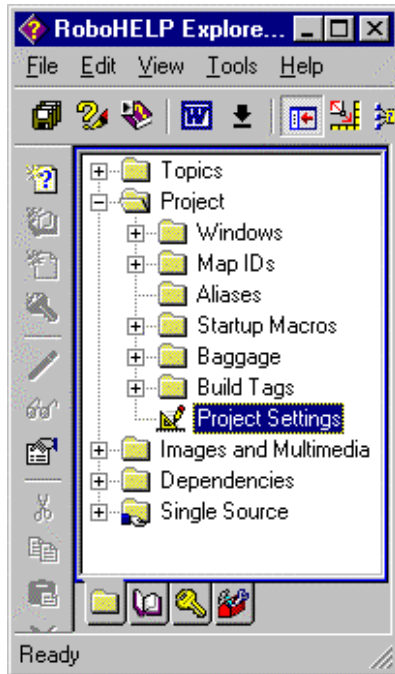


Setting Project Settings

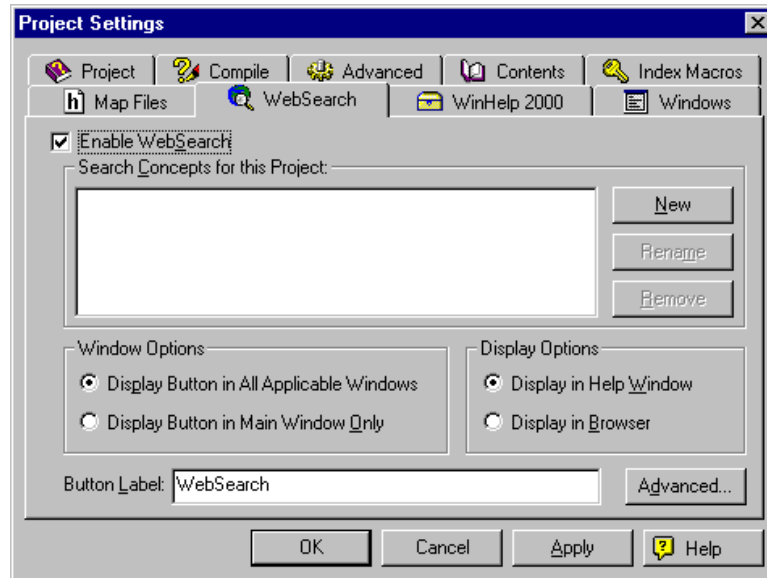
The **Project Settings** option within the **Project** tab allows you to change the settings associated with your Help project. These settings (or properties) define the characteristics of the project and make up each section in your project file (.HPJ). RoboHELP's Project Settings dialog provides several tabs for you to set the properties of your Help project:

To Set the Help Project Settings

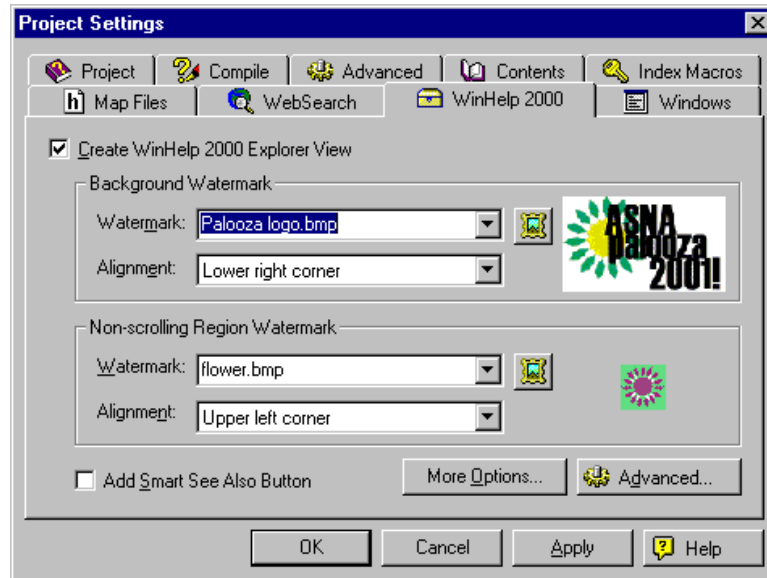
- Select the **Project - Project Settings** folder.



2. Select the **WebSearch** tab and select **Enable WebSearch**



3. Select the **WinHelp 2000** tab.



4. Select the following Watermarks:


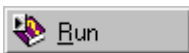
Background Watermark	Palooza logo.bmp	Lower right corner
Non-scrolling Watermark:	flower.bmp	Upper left corner

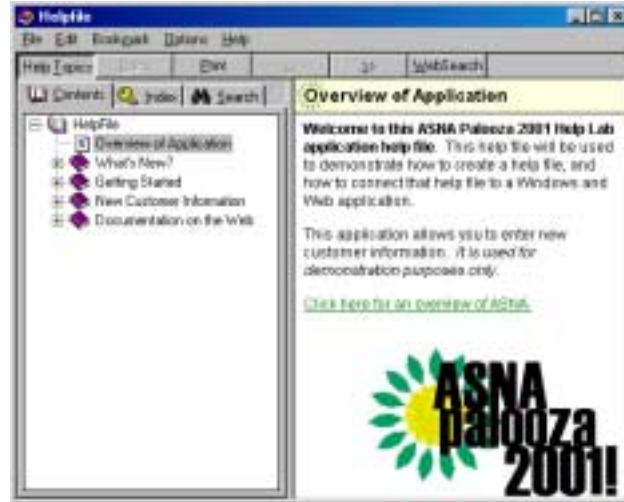
5. Select the other tabs. The information gets stored in individual **sections** in the **.HPJ** file.

Compiling the Help Project

Let's compile the help file again and look at some of the changes we have made to our help file.

To Compile a Help Project

1. Select **File – Compile** from RoboHelp Explorer, **Ctrl+M**, or select the Compile icon  in Microsoft Word.
2. Select the **Run** button  to view the Help File.
3. View all components of the help file, similar to the following.



Creating a Map File

One way in which context-sensitive help can be called from AVR is with the **HelpContextID** or **WhatsThisHelpID** properties. These properties require a **number** that correlates to a particular topic in the help file. This number resides in a **MAP** file (.HH) file.

Map files are text files that contain the list of Map IDs. Map files list, line by line, Topic ID and Map numbers. The Map file translates the Map number into the topic so the WinHelp engine can locate and display the appropriate context-sensitive Help topic when called by the application. RoboHELP automatically lists Map files in the [MAP] section of the Help project file (HPJ) whenever you create a new Map file or import an existing Map file.

The MAP file appears like the following entry, for each help topic:

```
#define Topic_ID    Map number
```

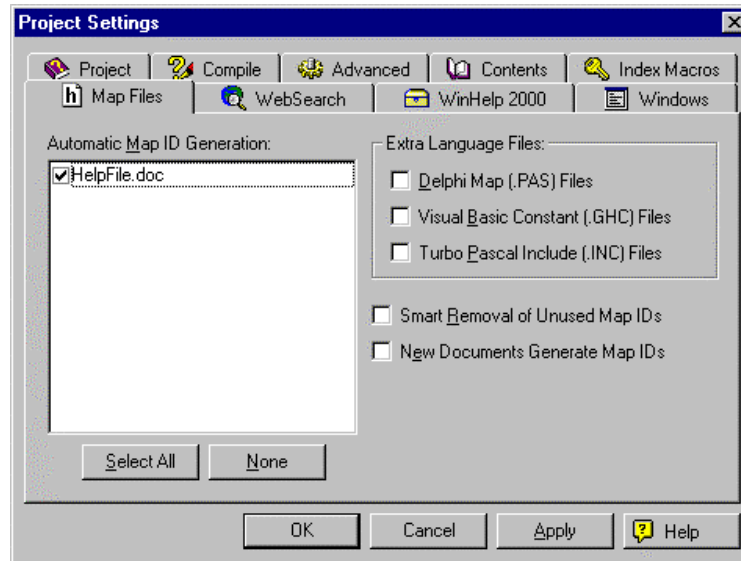
RoboHelp automatically generates a MAP file by automatically assigning **each** help topic in the help file a number, starting with **1** (which is the default).

The Help author and the Programmer both have the ability to generate the Map files that contain Map IDs. Since you both need to share the same Map IDs and Map files, it's a good idea to know who will create them for your project. In fact, it's best to sit down with your Programmer and plan a context-sensitive help strategy before either of you begin developing.

The **only purpose** of the MAP file is to contain those topics that you want to call from your application by pressing **F1** or the **WhatsThisHelp** button. Since the help file contains many other topics, such as overview, how-to's, etc, you may want to modify the .HH file. You can delete any topics you wish and renumber the existing topics as you want. You may even devise a numbering system, in which for example, all controls are the 100's, properties, 200's, etc, etc,. Once you change the file, you **must then** go back into RoboHelp and **turn off Automatic Generation**. Otherwise, the next time the help file is compiled, it will update the file, again adding a number for all of the topics, and the numbers will change and may not work correctly from your application.

To Create a Map File

1. The selection to create a MAP file was already selected when we inserted an existing file and **Converted** it to a RoboHelp file. You can verify by selecting the **Project** icon – **Project Settings** – **Map Files** tab.



2. **Compile** the help file again (**File – Compile**, or **Ctrl+M**).
3. Within the Word Document side, select **File – Open – Helpfile.HH**. The text file will appear as listed below. (Note that not the entire file is shown below).

```
define>Welcome → 1¶
define/Overview_of_Application→ 2¶
define/What_s_New_ → 3¶
define/Getting_Started → 4¶
define/New_Customer_Information→5¶
define/Cust_No_ → 6¶
define/Name → 7¶
define/Address_1 → 8¶
define/Address_2 → 9¶
```

4. You can **modify** the file, leaving only those topics that you want to access from your AVR or Web application. Delete the topics that are not needed, leaving only **15** main topics, as shown below.

```

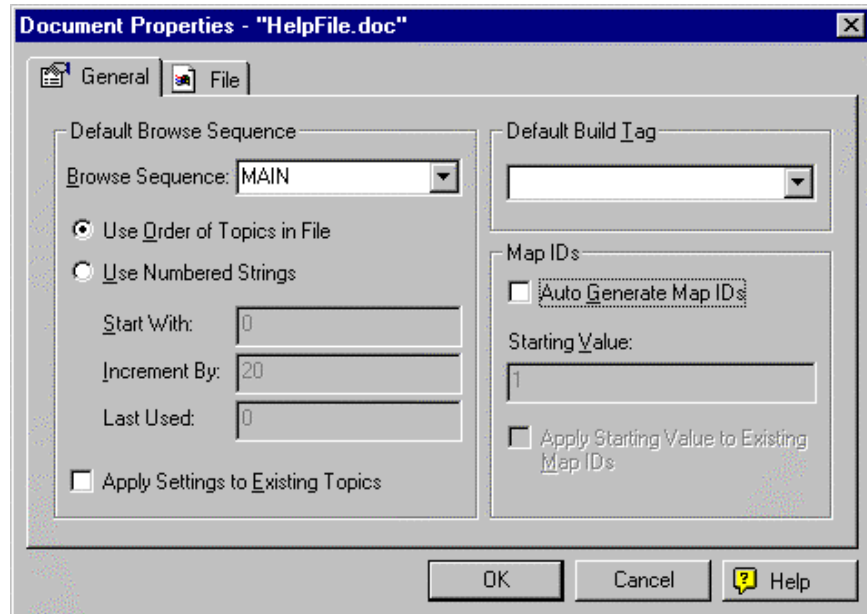
define·New_Customer_Information+19¶
define·Cust_No_ → 2¶
define·Name → 3¶
define·Address_1 → 4¶
define·Address_2 → 5¶
define·City → 6¶
define·State→ 7¶
define·Zip → 8¶

define·Whats_This_Help_CustNo → 9¶
define·Whats_This_Help_Name → 10¶
define·Whats_This_Help_Address1+11¶
define·Whats_This_Help_Address2+12¶
define·Whats_This_Help_City → 13¶
define·Whats_This_Help_State → 14¶
define·Whats_This_Help_Zip → 15¶

```

*Note: To save time, you can copy the **Helpfile.HH** file from \Session 05 – Creating Help Files\Examples\Help_File\Source Files folder into the current help project folder.*

5. Go to **RoboHelp – Document Properties** and turn off **Auto Generate Map ID's** and select **OK**.



6. Recompile the help file by selecting **File – Compile**, or **Ctrl+M**.

This concludes the creation of the Windows help file. Next, we will generate a 'WebHelp' file so that it can be used from a Web application.

What is WebHelp?

Web-based applications are becoming more and more sophisticated, so it is important to offer online Help assistance with any Web-based application.

WebHelp is a superset of HTML Help, and utilizes either Java or the Microsoft HTML Help ActiveX control to provide the cross-platform and browser independent functionality. WebHelp dynamically detects which browser is installed on the end-users system, and uses the appropriate control code to ensure the help system is optimally displayed. WebHelp is an **uncompiled html format** that supports standard Help features including a table of contents, index and full-text search, as well as context-sensitive Help. Since WebHelp is not a compiled format, WebHelp files can be stored on individual computers and Web sites, or viewed over a server, making it a good option for Web site or Web application Help files.

WebHelp gives you the ability to provide online Help for applications that run on the Web. Because of its configurable HTML and Dynamic HTML-based technology and browser-independent capabilities, WebHelp is a good choice for creating Help systems for the Web. WebHelp files can be used with any Web-based applications and can be viewed in both Internet Explorer and Netscape Navigator.

WebHelp can be run on Windows, Macintosh and UNIX operating systems, as well as on Intranet and Internet sites. WebHelp files can also be used with Web-based applications and can be viewed in both Internet Explorer and Netscape Navigator.

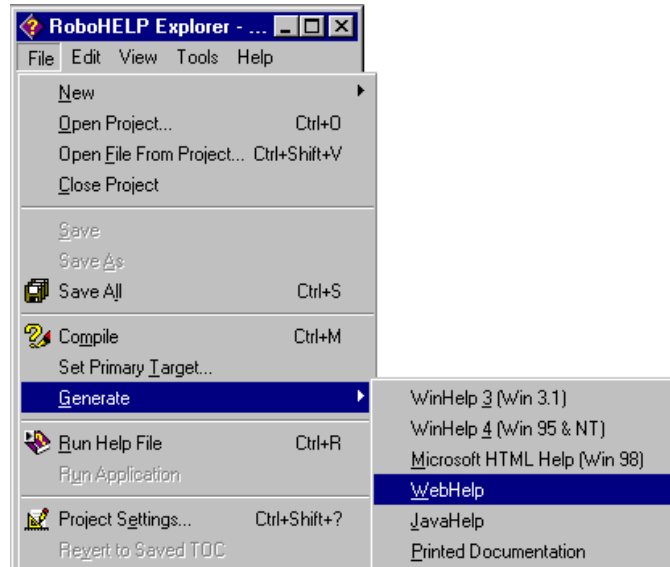
When users start your WebHelp project, the key components auto-detect the end-users' system (platform and browser) and launch the appropriate files for that particular system. This automatic process assures that your WebHelp project will look exactly the way you designed it, regardless of the browser or platform it is accessed from.

The manner in which WebHelp is delivered to the end user is determined by how WebHelp will be used. In the case of cross-platform application Help (i.e., user assistance for Windows, Macintosh, and UNIX-based applications), the contents of the WebHelp output folder are installed on the end user's computer system along with the application itself. When using WebHelp as user assistance for Web-based applications, or when using it to provide the navigation for an Intranet-based information system, the WebHelp output folder is placed on a server, and the files are accessed individually from the end user's browser, just like normal Web pages.

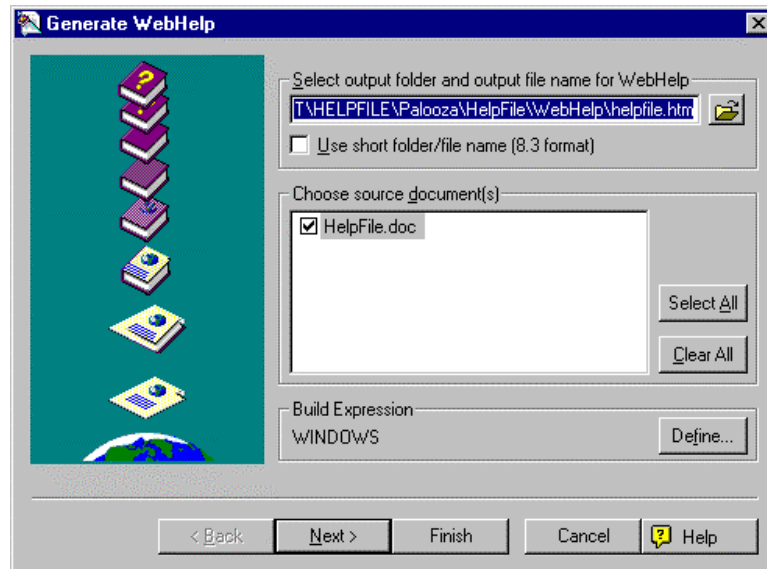
Generating WebHelp

To Generate a WebHelp File

1. Select **File – Generate – WebHelp**.

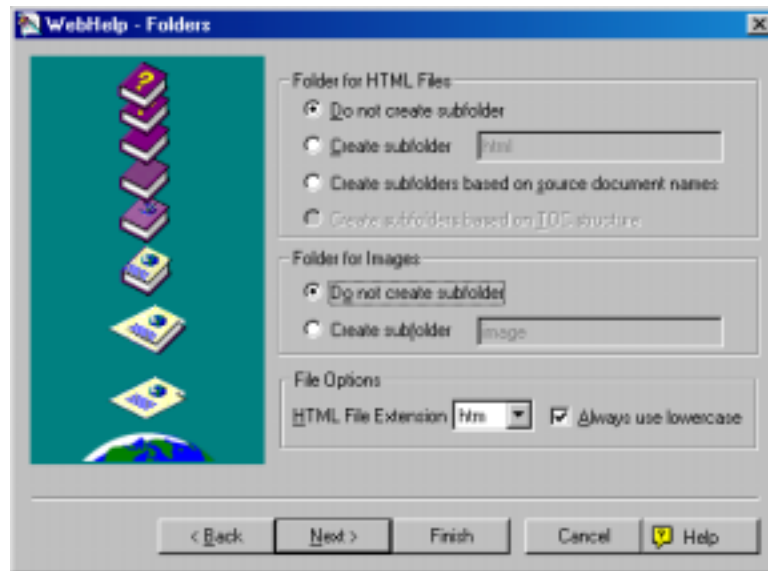


The following dialog will display.



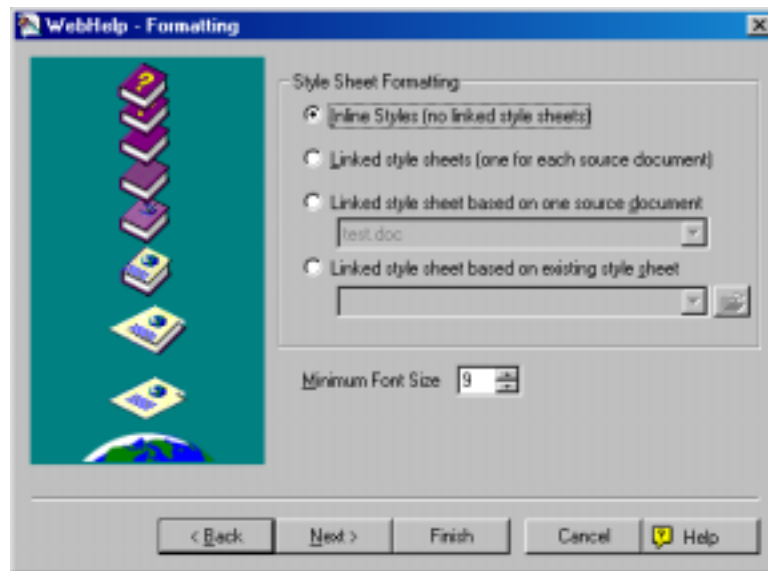
By default, a folder called **WebHelp** will automatically be generated within your current project folder. All of the files and folders generated will be placed within that folder.

3. Select **Next >**. The following dialog will display.



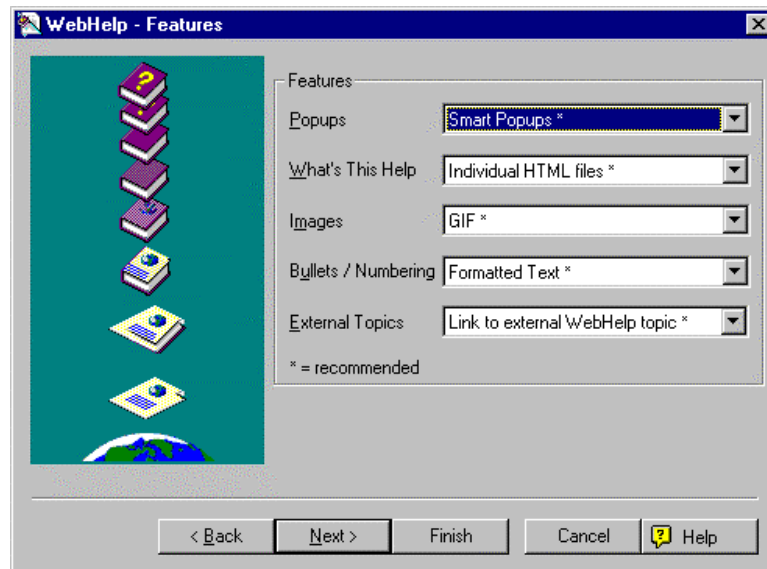
This screen enables you to specify if you want separate folders to hold your images, or to divide up your HTML files. For this small example, let's just keep everything in one folder.

4. Select **Do not create subfolder** for both **HTML** files and **Images**, then select **Next >**. The following dialog will display.



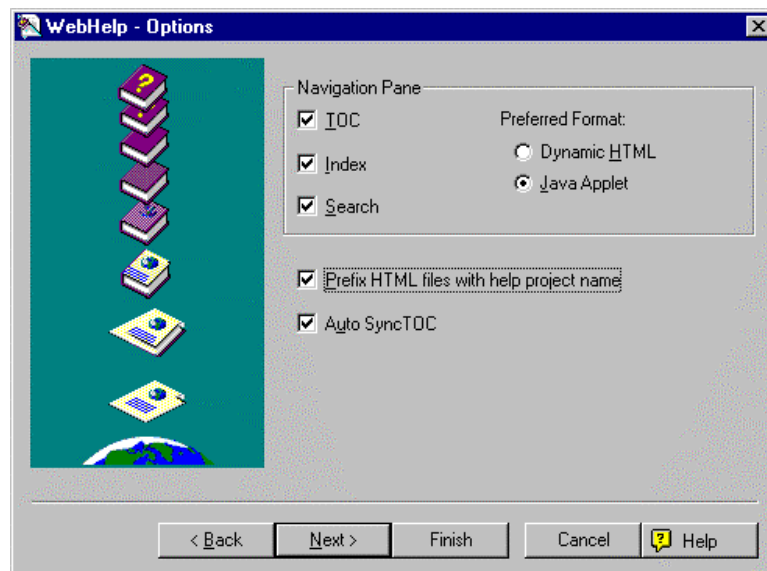
This dialog specifies how you want to format the HTML topic files. You have the option of working with or without style sheets. If you don't want to use style sheets, the HTML topics will be formatted using inline styles (which is like applying manual character and paragraph formatting in Word). If you want to use style sheets, you can create new style sheets based on the formatting in the WinHelp topics or you can select a style sheet you already use with other HTML topics.

4. For this example, we will use the default, **Inline Styles**, so select **Next >**. The following dialog will display.

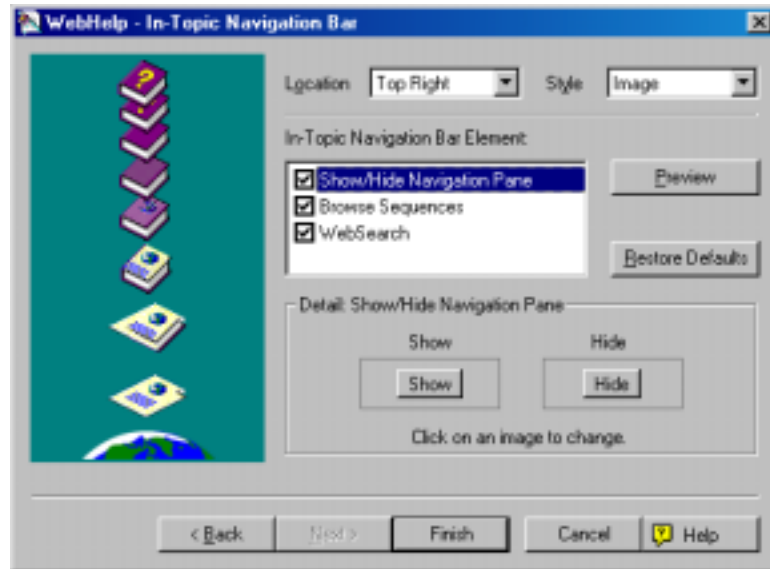


This screen specifies various features of the help file. The recommended default settings are selected.

5. Select **Next >** to continue. The following dialog will display.

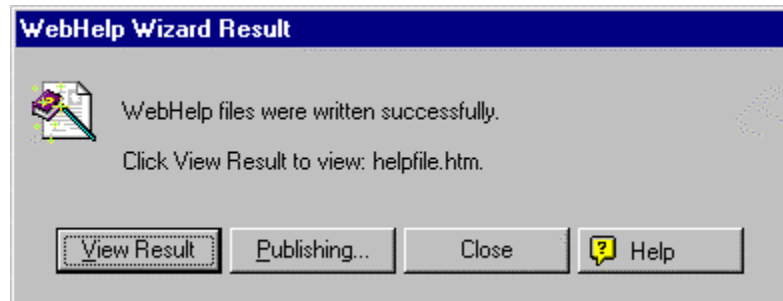


6. Select **all options** and change the selection from **Dynamic HTML** to **Java Applet**, and select **Next >**.

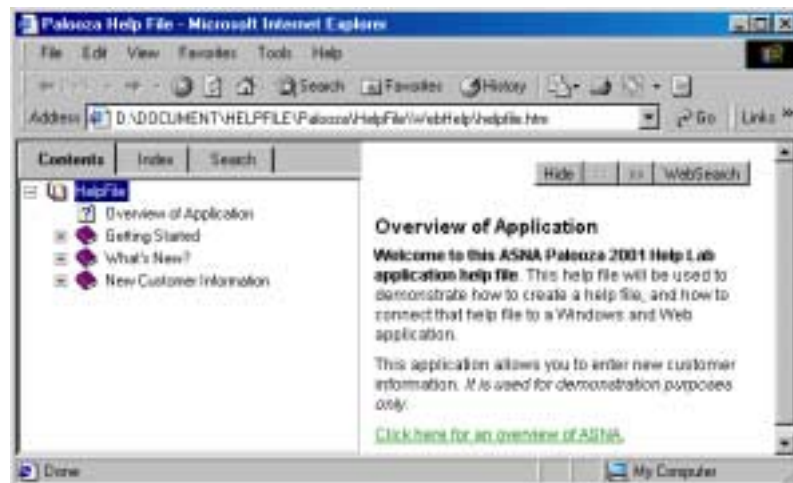


This screen controls the contents of the Navigation Bar. The navigation bar appears in the right-hand pane of your WebHelp project in a location you specify.

7. Select **all Navigation Bar elements** and select **Finish**. The following dialog will display.



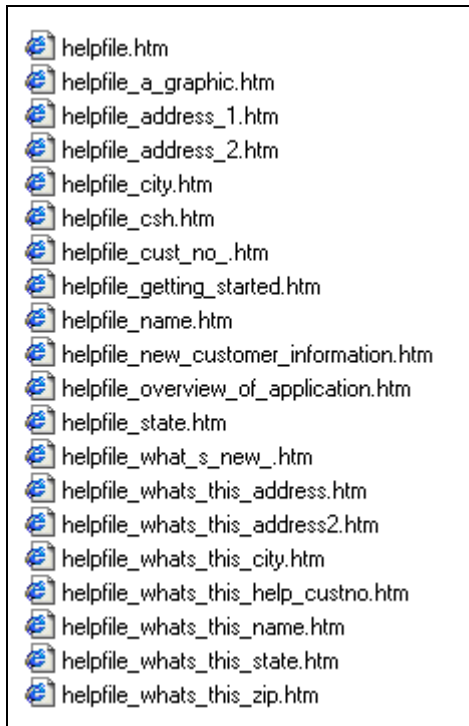
8. Select **View Result** to view the compiled HTML file titled **helpfile.htm**, as shown below.



9. To view all of the files that were generated, go to the folder in which your help file resides, and notice that there is a **WebHelp** folder located within that folder.

Note that each topic in your help file is now an **individual HTML file**. We will be using these individual HTML files that were generated when we incorporate help into our Web application in Step 3.

Some of the files are listed below. The **help file name** was added as a **prefix** to all of the file names, which defaults to the topic title.



Congratulations!!!

This completes the generation of our help file for both our Windows and Web application. We will use the help file and web help file generated in this section for the next chapters on connecting the help file to a Windows and Web application.



Step 2: Connecting a Help File to an AVR application

What you will learn in Step 2:

- Calling Help from a Help Menu.
- Calling Help from a Help Button using different Winhlp32 parameters.
- Calling Context-Sensitive Help using HelpContextID and HelpKey properties to a Main and Secondary help window.
- Calling Context-Sensitive Help using the WhatsThisHelp button and WhatsThisHelpID property.

Approximate Time to Complete Step 2:

Approximately 30 minutes to an hour.

What the AVR App will look like after completing Step 2:



What We Will Cover in Step 2

In this step, we are going to utilize the Help File created in Step 1 and incorporate it into our AVR application.

The Help File will be accessed from a Help Menu, accessing the entire help file. We will also utilize context-sensitive help from a Help button and from each IOField.

The application we will be using is shown below.

The screenshot shows a Windows application window titled "Connecting Help to an Application". The window has a menu bar with "File", "Edit", "View", "Tools", "Window", and "Help". The main area contains a header with "ASNA" logo, "New Customer Information" text, and a small image of a person. Below the header is a form with fields for "Cust No.", "Name/Address" (sub-section), "Name:", "Address 1:", "Address 2:", "City:", "State:", and "Zip:". At the bottom are "OK", "Cancel", and "Help" buttons.

The Help file already has the appropriate pieces in place in order to do the following:

- Call the entire Help file from the **Help menu**.
- Call a topic called New Customer information when the **Help button** is pressed.
- Display individual help topics using the **HelpContextID** and **HelpKey** properties, as well as display **WhatsThisHelp** popups using the **WhatsThisHelpID** property.

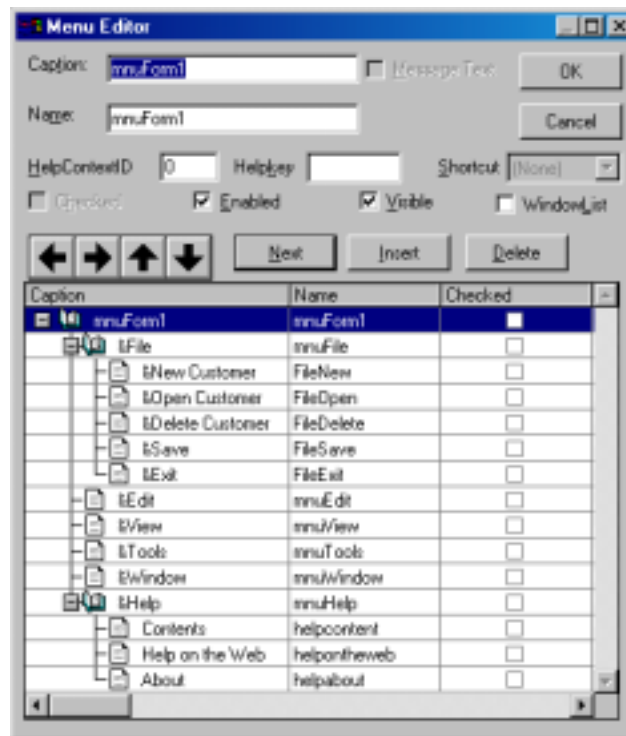
Calling Help from a Help Menu

You can call an entire Help File from a Help menu in code by calling **Winhlp32.exe** and the **name** of the help file. We are also going to call a WebHelp file that resides on the web.

The menu items are created using **Menu Editor**.

To Call a Help File from a Help Menu

1. Copy the **HelpFile.HLP** and **HelpFile.CNT** files from the Help File folder created in the previous section, or previous lab into the current folder: the **\Session 10 – Connecting Help Files to Apps\Examples\AVR_App**.
2. **Open** the AVR application in the **\Session 10 – Connecting Help Files to Apps\Examples\AVR_App** titled **Application.VRM**.
3. Select **Menu Editor** from the **Tools** menu and view the **Names** of the Help menu items that have been previously created, as shown below.



4. Go to the code editor and select **helpcontent** from the Controls box and **Click** from the Events box. Add 1 line using **OSEXEC** to call the Winhelp engine, (**Winhlp32**), followed by the name of the Help file in a string, as shown below.

```
BEGSR helpcontent Click
OSEXEC "Winhlp32.exe Helpfile.hlp"
ENDSR
```

5. **Run** the application.
6. Click on **Help - Contents**. The entire Help file will display.

You can also call the help file directly, without having to enter **Winhlp32.exe**, as the help file is already associated with the Winhlp32

engine. This time, remove **Winhlp32.exe**, leaving just the help file name in quotes, as shown below and **Run** the application again.

```
BEGSR helpcontent Click
OSEXEC "Helpfile.hlp"
ENDSR
```

Calling an HTML File Locally or on the Web

The **Help** menu also has a menu option titled **Help On the Web**. The menu item titled **helpontheweb** calls a **Webhelp** help file on ASNA's web site. Note that for this exercise during Palooza, we will use Internet Explorer to call an html file on our local system.

1. Go to the code editor and select **helpontheweb** from the Controls box and **Click** from the Events box.
Add 1 line using **OSEXEC** to call Internet Explorer, (**iexplore.exe**), followed by the name of the html file, as shown below.

```
BEGSR helpontheweb Click
OSEXEC "iexplore.exe \Session 05 – Creating Help Files\Examples
\Help_File\Source Files\default.htm"
ENDSR
```

2. **Run** the application.
3. Click on **Help – Help on the Web**. The local HTML file will display.

Note, to call an HTML file on the web, you would enter the path as HTTP:// followed by the URL, as in the following example. (You also do not need to enter iexplore.exe when HTTP protocol is used).

```
OSEXEC "http://support.asna.com/kb/documentation/help_files/avr31/
webhelp.htm"
```

Calling Help using a Help Button

Notice that a **Help** button has also been added to the sample application that will take you to the topic describing the entire screen, or form (**New Customer Information**).

Calling help from a help button is similar to calling a help menu item. The only difference is now you want to go directly to a **particular topic** in the help file.

Winhlp32 has some parameters that you need to pass, either using a **Map number** (-N), **Keyword** (-K) or **Help Topic ID** (-I).

In this section, we will call the help topic using each of these parameters, along with an example using the **CommonDialog** control.

To Call a Help File from a Help Button

Enter the following code for the **Help** button named **btnHelp**.

1. Go to the code window and create a **Click** event for **btnHelp**.

```
BEGSR btnHelp Click
ENDSR
```

2. Enter the following code (in bold), or copy and paste the **OSEXEC** line entered for the **HelpContent** item and copy into the subroutine, as shown below.

```
BEGSR btnHelp Click
OSEXEC "Winhlp32.exe Helpfile.hlp"
ENDSR
```

However, in this case, we want to call a **particular topic** in the help file, so we need to integrate context-sensitive help. Winhlp32 has some parameters you can pass to indicate the topic you would like to go to in the help file.

Refer to page 57 for a listing of all of the Winhlp32 parameters.

For this step, we will use the ‘**MAP**’ number generated for the Help Topic called “**New Customer Information**”. A MAP number is noted by the **-n** parameter, followed immediately by the associated map number. If we go back to our MAP file (.HH file), we will see that the **New_Customer_Information** Help Topic was assigned the number **1**.

Using the Map Number Parameter

3. To specify the **New_Customer_Information** help topic in code, use the **-n1** parameter **between** Winhlp32.exe and the name of the help file, as shown below.

```
BEGSR btnHelp Click
OSEXEC "Winhlp32.exe -n1 Helpfile.hlp"
ENDSR
```

4. **Run** the application and select the **Help** button. Note that the help topic titled “New Customer Information” will display.

*Note that a MAP file does not have to be created, and that there are several ways to call an individual topic from a help button instead of using the **-n** parameter.*

- You can use the **-K** parameter to indicate a keyword, but the keywords **cannot contain any blanks**. For example, you could use, **-KName**, but *not* **-KNew Customer Information**.

*To get around this, you can either use the **-I** parameter, or use the Help properties associated with the Common Dialog Control.*

- You can also use the **-I** keyword, which contains the name of the **help topic ID**. For instance in this example, the topic ID for New Customer Information is **New_Customer_Information**. Note that topic ID's will always contain an underscore for spaces between words, i.e., **-New_Customer_Information**.
- You can also use the **CommonDialog** control, which has a **HelpContextID** property to call a number (same as **-n**), or the **HelpKey** property to call a keyword (same as **-K**, except that it accepts a string that can contain spaces).

*To summarize, if you are calling a help topic that does not have a map number, and all of the keywords specified contain a blank, then you must either use the Winhlp32's **-I** keyword followed by the help topic ID, or use the **CommonDialog**'s **HelpFile**, **HelpCommand**, **HelpKey** and the **ShowHelp** method.*

Using the Help Topic ID Parameter

In this section, we will call the New Customer Information help topic again using the **-I Winhlp32** parameter.

5. Change the **-n** parameter to a **-I**, followed by the help topic ID for the topic New Customer Information, which is **New_Customer_Information**, as shown below.

```
BEGSR btnHelp Click
OSEXEC "Winhlp32.exe -INew_Customer_Information Helpfile.hlp"
ENDSR
```

6. **Run** the application and select the **Help** button. Note that the help topic titled "New Customer Information" will again display.

Using CommonDialog to Display a Help Topic by Keyword

In this section, we are going to use the **CommonDialog** control's **Help** properties to display a help topic by a **keyword**. By using the **CommonDialog**'s **HelpKey** property, you can specify a keyword containing blanks.

Refer to the AVR help on **CommonDialog** for a listing of all its help properties that are available.

To Use the Common Dialog Control in Conjunction with a Keyword

1. Add a **CommonDialog** control onto the form.
2. Go to the **Code** window and either delete or **comment** out the previous code for the **btnHelp** command button, and add in the following code for **btnHelp** using the **Common Dialog** control.

```
BEGSR btnHelp Click
CommonDialog1.HelpFile = "HelpFile.hlp"
CommonDialog1.HelpCommand = 7
CommonDialog1.HelpKey = "New Customer Information"
CommonDialog1.ShowHelp
ENDSR
```

Refer to the AVR Help file for more information on each property. Note that the **HelpCommand =7** specifies a **Help Key** and must be followed by the **HelpKey** property.

3. **Run** the application and select the **Help** button. Note that again, the New Customer Information topic will display.

Winhlp32 Parameters

You will probably want users to access your Help File from a **Help Menu** included in your application. This involves creating the Menu controls by using Menu Editor, then using the OSEXEC command to execute the help file, using WINHLP32, specifying any additional parameters as needed.

If you want to call a particular topic from a Help button included on your form, you use OSEXEC/EXEC, but you also need to pass a **Winhlp32 parameter** to identify the individual topic in which you want to display.

The following is a listing of the WINHLP32 parameters.

Winhlp32.exe Parameters

```
winhlp32.exe [[-H] [-G[n]] [-W window-name] [-K keyword]
[-N context-num] [-I topic-id] [-P pop-up-id] HLP-
filename]
```

The following parameters may be used when starting Winhlp32:

<u>Parameter</u>	<u>Description</u>
-G[n]	Creates a configuration (.gid) file and quits. If a number is specified, it determines which extensible tab to display by default the first time the Help file is opened. A value of 1 would be the first tab beyond the Find tab.
-H	Displays the Winhlp32.hlp Help file.
-I topic-id	Displays the Help topic with the specified the topic ID.
-K keyword	Displays the topic identified by the specified keyword. <i>The keyword cannot contain any blanks.</i>
-N context-num	Displays the topic specified by the context number (defined in the [MAP] section of the project file).
-P pop-up-id	Displays the specified pop-up topic. You must use the -P switch in combination with the -I or -N switch, as shown in the following examples: WINHLP32 -P -I EXEC_WINHELP HCW.HLP WINHLP32 -P -N 311 MYFILE.HLP
-W window-name	Displays the topic in the specified window definition.
HLP-filename	Specifies the Help file to display. If a name is not specified, the File Open dialog box appears.

Examples:

The following examples show how you would call a particular topic in a help file from a **help button** using the **Winhlp32 parameters** in Caviar.

The following displays the help topic identified by **Context number** 12 (in the map file) in "Myhelp".

```
OSEXEC "Winhlp32.exe -n12 Myhelp.hlp"
```

The following displays the help topic identified by the **Keyword** "Name" in "Myhelp".

```
OSEXEC "Winhlp32.exe -kName Myhelp.hlp"
```

The following displays the help topic identified by the **Topic ID** called "New_Customer_Information" in "Myhelp".

```
OSEXEC EXEC "Winhlp32.exe -INew_Customer_Information
Myhelp.hlp"
```

The following examples show how to display the help topic identified by either a **Keyword** or **Topic ID** in a **Popup** window in "Myhelp".

```
OSEXEC EXEC "Winhlp32.exe -P -KName Myhelp.hlp"
OSEXEC EXEC "Winhlp32.exe -P - INew_Customer_Information
Myhelp.hlp"
```

If you will not be calling the help topic using a number from a map file (-n), and your keyword(s) for the help topic have spaces, then you must either use the -I parameter, or the CommonDialog control's help properties.

Creating Context-Sensitive Help from an Application

So far, we have used AVR code to call an entire help file, and, by passing a Winhlp32 parameter, we have called a particular topic within that help file.

In this section, we will adding **Context-sensitive** help to the IOFields. There are basically only two steps you must perform within Visual RPG. There is no additional coding that needs to be done.

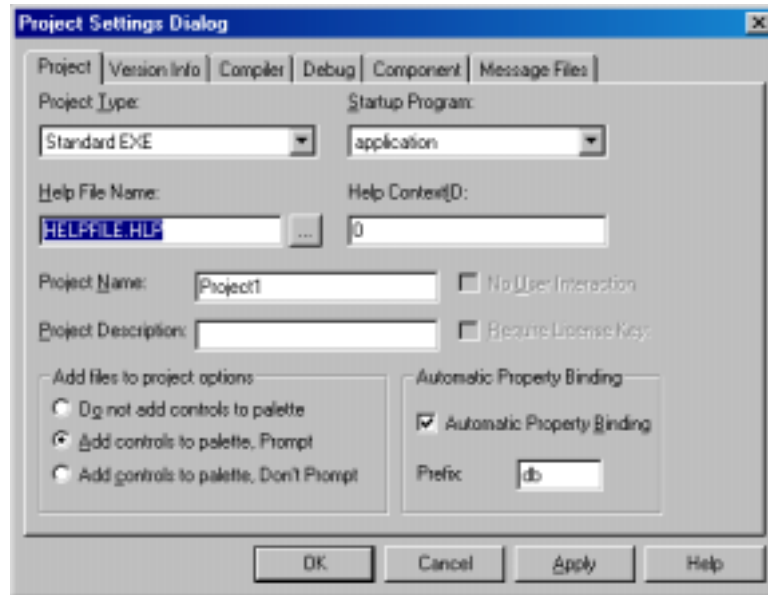
1. Specify the **name of the Help File** created for the application in **Project - Project Settings**.
2. Assign the appropriate **HelpContextID** or **HelpKey** created in the Help file to the desired fields using either the **HelpContextID**, **HelpKey**, or **WhatsThisHelpID** properties.

Specifying the Help File for the Project

You must first tell the AVR application what the name and location of the Help file for this application is. You specify the name of the Help file in **Project - Project Settings – Project** tab.

To Specify a Help File for the Project

1. Select **Project - Project Settings**.
In the **Project** tab, there is a field for you to enter the Help file name for the application.
2. Enter the help file name of **Helpfile.hlp**, or click the button to the right to browse for the file.



3. Select **A**pply and **O**K when finished.

Assigning Context-Sensitive Help in the Application

Once the help file is specified for the project, you are ready to begin using the **HelpContextID**, **HelpKey** or **WhatsThisHelp** properties.

In this section, we are going to be using the **HelpContextID** and **HelpKey** properties.

- You call topics using the **HelpContextID** property by specifying an associated 'MAP number' that is contained in the .HH file.
- You call topics using the **HelpKey** property by specifying an associated 'Keyword' for that topic.

Using the HelpContextID Property

To Use the HelpContextID Property

1. Open or view the **MAP** file assigned to the help file (**Helpfile.HH**). In this section we will **only** be using the following map numbers.

#define Cust_No_	2
#define Name	3
#define Address_1	4
#define Address_2	5
#define City	6
#define State	7
#define Zip	8

2. Select the **Cust No** IOField and enter the number **2** into the **HelpContextID** property.

3. Select the rest of the IOFields and enter their corresponding Map number into the **HelpContextID** property.

IOField	HelpContextID
Name	3
Address 1	4
Address 2	5
City	6
State	7
Zip	8

4. **Run** the application and **press F1** into each of the **IOFields**. The following will display when **F1** is pressed in **Cust No**.



Note that Winhlp32 is launched, the entire Help Window is displayed, and the Contents tab and topic are displaying the associated topic assigned to that map number.

Using the HelpKey Property

If you do not want to generate a MAP file, you can use the **HelpKey** property. In the HelpKey property, you enter a **keyword string** associated with that topic. Note that you may have **many** keywords assigned to a topic, but you only need to specify one, and it doesn't matter which one, and blanks are accepted.

AVR looks at the HelpContextID property first (as long as WhatsThisHelp is not enabled), so we also need to select the HelpContextID and enter 0.

To Use the HelpKey Property

1. Select the **Cust No** IOField and enter the keyword **Cust No** into the **HelpKey** property.
2. Select the **HelpContextID** property and enter a 0.
3. Enter the following **Keywords** for each of the following help topics.

IOField	Help Keyword
Cust No	Cust No
Name	Name
Address 1	Address 1
Address 2	Address 2
City	City
State	State
Zip	Zip

4. Select the **HelpContextID** property and enter a 0 for each IOField.
5. **Run** the application and **press F1** into each of the **IOFields**.

Note that when Winhlp32 is launched, the entire Help Window is displayed, and the Contents tab and topic are displaying the associated topic assigned to that keyword.

Enabling Secondary Windows

In Help files, there is one **Main** window. The **Main** window is the default window. All topics automatically display in the Main window unless you specify otherwise. You can create other custom windows called **Secondary** windows. You can specify the appearance, and attributes, and options of the both windows to make it fit into your overall Help system design.

A secondary window is specified by indicating a > **sign** followed by the **name** of the **secondary window** to display, either within the help or when specifying the help file within AVR.

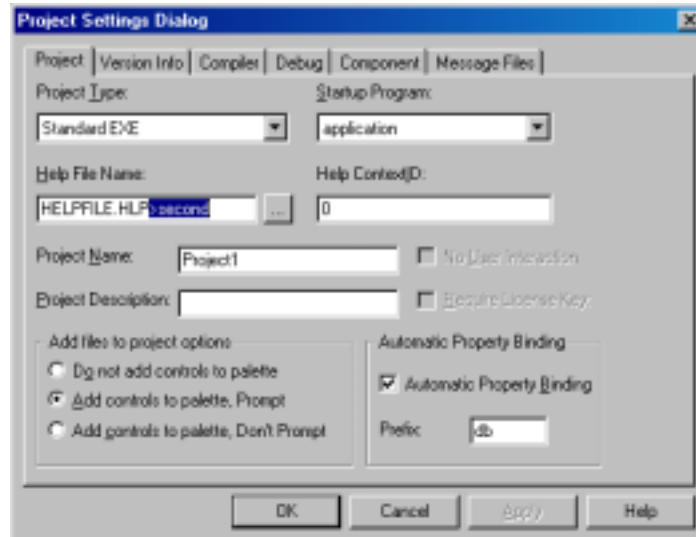
Note that there is a secondary window enabled by default within any help file called **Second**.

In this section, we will specify that the help topics display in a secondary window for the application.

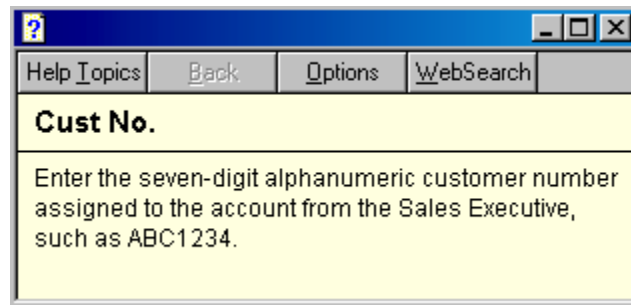
*Note that secondary windows are only enabled for the **HelpContextID** and **HelpKey** properties, and is ignored when using the **WhatsThisHelpID** property.*

To Specify a Secondary Window for the Help Project's HelpContextID and HelpKey Properties

1. Go back into **Project – Project Settings** and specify the Secondary window called “**second**” by entering a **>second** after the name of the help file, as shown below.



2. **Run** the application.
3. Press **F1** in **Cust No.**
4. Notice this time that the Help topic displays in a “**smaller**” window with the specification set for the Secondary window in the help file as far as position, size, options, and color, as shown below.



Even though Winhlp32 is launched, you do not see the “**Contents**” tab for the help file. This is a convenient way to display the topics, since if the user wants to view more information in the help file, they only need to select the **Help Topics** button.

Also, the height of the secondary window automatically fits the size of the text, so the window is smaller than displaying the text to the Main window.

Using the WhatsThisHelpID Button and Property

In this section, we will use the **WhatsThisHelpID** property and **WhatsThisHelp** button to display context-sensitive help in a ‘popup’ window.

Note that if you want to use WhatsThisHelp popups, a MAP file must be generated for the specified help file, as there is not a property to specify a keyword.

To Use the WhatsThisHelp Property

1. Be sure to specify the name of the help file for the application by selecting the **Project** tab within **Project – Project Settings**.

Note that you can still leave the **>Second** after the help file to specify a secondary window. It will not affect the displaying of WhatsThisHelp popups, as it will be ignored.

2. Next, get the “**whats this help**” topic **context-id numbers** assigned to the help topics from the help authoring person, or from the **MAP** file (.HH) that was generated by the Help file.

The following are the Map numbers we will use in this section in conjunction with the **WhatsThisHelpID** property.

```
#define Whats_This_Help_CustNo      9
#define Whats_This_Help_Name        10
#define Whats_This_Help_Address1    11
#define Whats_This_Help_Address2    12
#define Whats_This_Help_City        13
#define Whats_This_Help_State        14
#define Whats_This_Help_Zip         15
```

Property Settings for the Form

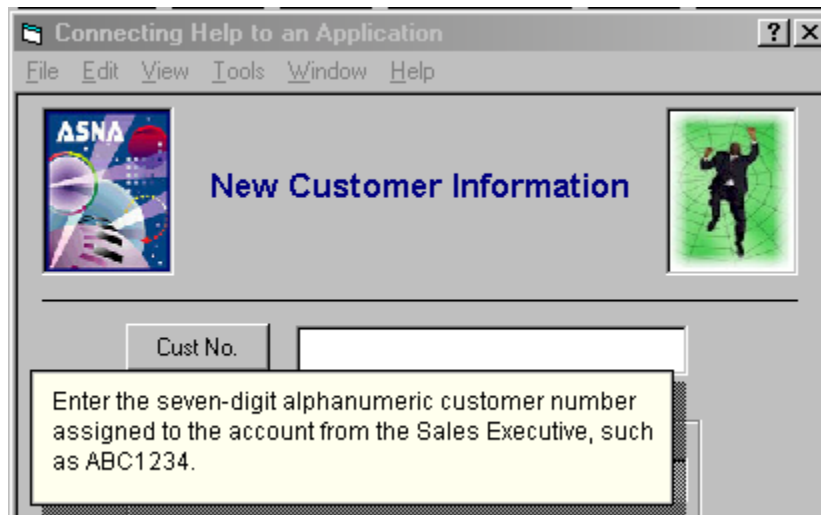
3. There are several **form** properties that must be set to enable AVR to use **WhatsThisHelp**, along with the **WhatsThisHelp** button. Click on the form, and set the following properties as listed below.

Property	Setting
BorderStyle	Use 1, 2 or 3
MaxButton	Set to False
MinButton	Set to False
WhatsThisButton	Set to True
WhatsThisHelp	Set to True

4. Select the **Cust No** IOField and enter the number **9** into the **WhatsThisHelpID** property.
5. Select the rest of the IOFields and enter their corresponding Map number into the **WhatsThisHelpID** property, as listed below.

IOField	HelpContextID
Name	10
Address 1	11
Address 2	12
City	13
State	14
Zip	15

- Run the application.
- Click on the **WhatsThisHelp** button in the **Form's Title Bar**, then click in the **Cust No.** IOField. The WhatsThisHelp topic for Cust No will display in a popup window, as shown below.



- Click on the other fields, and their corresponding information will display in a pop-up window.

Note that you can also select the F1 key in the IOField, and the WhatsThisHelp popup will display as well.

Note that if the WhatsThisButton is used, the Form's Minimize and Maximize buttons are not enabled. You can still utilize the WhatsThisHelp popup capability without using the WhatsThisHelp button. To do this, simply set the WhatsThisButton button to false for the form, but keep WhatsThisHelp to true.

- Click on the form and set the **WhatsThisButton** property for the Form to **False** and set the **MinButton** and **MaxButton** properties to **True**.

10. Run the application again.

Notice that the Minimize and Maximize buttons are enabled, and that there is no Help button. However, WhatsThisHelp is still enabled, so AVR will launch the number in the WhatsThisHelpID properties when F1 is pressed, and will ignore if there is an entry in the HelpContextID or HelpKey properties.

Congratulations!

You have just completed the process to connect a Help file to an AVR application.

Continue with the next Step for the instructions to connect a WebHelp file to a similar Web application.

This Page Intentionally Left Blank



Step 3: Connecting WebHelp to a Web Application

What you will learn in Step 3:

- Calling a WebHelp file from a Help Button or Link into another browser window using JavaScript.
- Calling Context Sensitive Help into another browser window using JavaScript.

Approximate Time to Complete Step 3:

1 hour.

What the Web site will look like after completing Step 3:

The screenshot shows a web application interface with a yellow background. At the top, there is a navigation bar with buttons for Home, Products, Services, Downloads, Contact Us, and Help. Below the navigation bar, there is a header area with a logo on the left and a small image of a person on the right. The main content area is titled "New Customer Information". It contains a form with the following fields and controls:

- Cust No:** A text input field with a question mark icon to its right. Below it are "Previous" and "Next" buttons.
- Name:** A text input field with a question mark icon to its right.
- Address 1:** A text input field with a question mark icon to its right.
- Address 2:** A text input field with a question mark icon to its right.
- City:** A text input field with a question mark icon to its right.
- State:** A dropdown menu currently showing "Alaska" with a question mark icon to its right.
- Zip:** A text input field with a question mark icon to its right.

At the bottom of the form, there are "Submit" and "Reset" buttons.

What We Will Cover in Step 3

In this step, we are going to utilize the WebHelp files created in Step 1 and incorporate them into our Web application.

The WebHelp file will be accessed from a Help button, accessing the entire help file, as well as utilizing context-sensitive help by selecting an icon next to each field.

Calling the WebHelp file is basically the same as creating a hyperlink to our help files' main .HTM file. The only difference is that we will want to display the file into a separate browser window and will use **JavaScript** code to do this. For a great Web site on JavaScript, see <http://www.w3scripts.com>.

The Web application we will be using is shown on the previous page.

What is WebHelp?

Web-based applications are becoming more and more sophisticated, so it is important to offer online Help assistance with any Web-based application.

WebHelp is a superset of HTML Help, and utilizes either Java or the Microsoft HTML Help ActiveX control to provide the cross-platform and browser independent functionality. WebHelp dynamically detects which browser is installed on the end-users system, and uses the appropriate control code to ensure the help system is optimally displayed. WebHelp is an **uncompiled html format** that supports standard Help features including a table of contents, index and full-text search, as well as context-sensitive Help. Since WebHelp is not a compiled format, WebHelp files can be stored on individual computers and Web sites, or viewed over a server, making it a good option for Web site or Web application Help files.

WebHelp gives you the ability to provide online Help for applications that run on the Web. Because of its configurable HTML and Dynamic HTML-based technology and browser-independent capabilities, WebHelp is a good choice for creating Help systems for the Web. WebHelp files can be used with any Web-based applications and can be viewed in both Internet Explorer and Netscape Navigator.

WebHelp can be run on Windows, Macintosh and UNIX operating systems, as well as on Intranet and Internet sites. WebHelp files can also be used with Web-based applications and can be viewed in both Internet Explorer and Netscape Navigator.

When users start your WebHelp project, the key components auto-detect the end-users' system (platform and browser) and launch the appropriate files for that particular system. This automatic process assures that your WebHelp project will look exactly the way you designed it, regardless of the browser or platform it is accessed from.

The manner in which WebHelp is delivered to the end user is determined by how WebHelp will be used. In the case of cross-platform application Help (i.e., user assistance for Windows, Macintosh, and UNIX-based applications), the contents of the WebHelp output folder are installed on the end user's computer system along with the application itself. When using WebHelp as user assistance for Web-based applications, or when using it to provide the navigation for an Intranet-based information system, the WebHelp output folder is placed on a server, and the files are accessed individually from the end user's browser, just like normal Web pages.

Calling the Entire WebHelp file from a Help Button

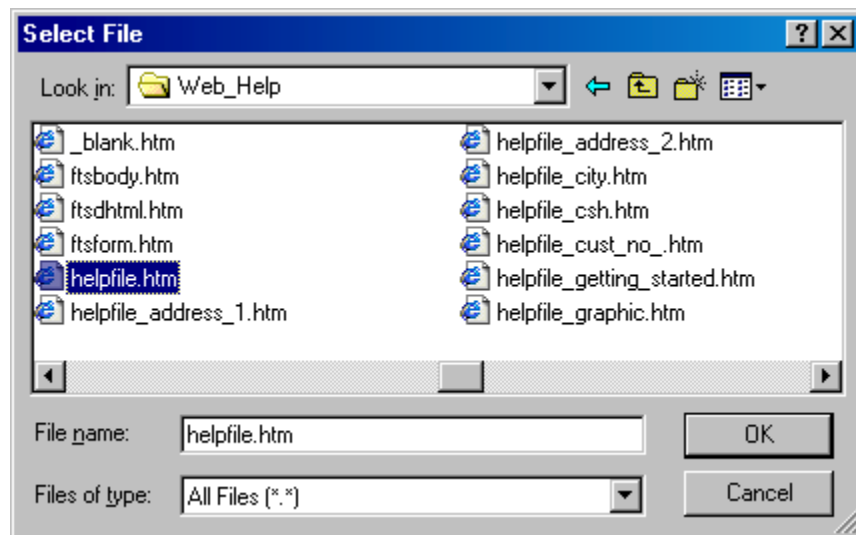
First, we will call an entire WebHelp file (**an html file**) from a Help button by merely adding a **Hyperlink** to the html file.

Then, in the next step, we will actually add a JavaScript variable and function to display the help file in a separate browser window. (If the help is not displayed in a separate window, then 'closing' the help file will 'close' the application).

We will use FrontPage 2000 as our HTML editor. However, you could also use your editor of choice, but note that they may be some differences in the steps to follow.

To Call a Help Web Help File from a Help Button

1. Open **FrontPage 2000**.
2. Open the HTML file titled **Application**.
3. Click on the **Help** button and select **Insert – Hyperlink**. Click on the browse button **and** select **helpfile.htm**.

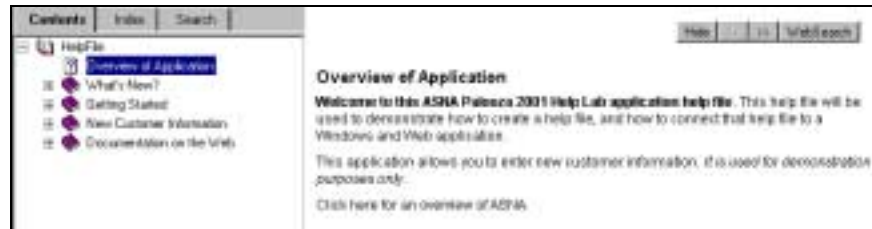


4. Select **helpfile.htm** in the **\Session 10 – Connecting Help Files to Apps\Examples\Web_Help** folder and select **OK**.

If you look at the HTML code, an **<a href>** tag was added to the image **help button.gif** pointing to the **helpfile.htm** file.

```
<a href="Examples/Web_Help/helpfile.htm"><IMG
border=0 height=25 src="help%20button.gif"
width=112></a></td>
```

5. Save the File by selecting **File – Save**.
6. Select the **Preview** tab and click on the **Help** button. The entire help file will display, as shown below.



Notice that the help file displays in the same Window as the application, so when you close the help file, you are actually closing the application.

We can define a separate window to display the help file in. We will create a variable **StrHelpOptions2** that defines the window parameters for calling the entire help file.

See the following step for the steps in defining a separate window.

Displaying the Entire Help File in a Separate Window

First, we need to define the Window parameters for calling the full Webhelp help system. We will define a variable called **strHelpOptions2** that merely defines the **window parameters**.

Next, we will use the JavaScript **window.open** function (called ShowHelp2) that opens a window with the strHelpOptions2 variable settings.

Lastly, we will update our Help button hyperlink to call the JavaScript ShowHelp2 function, which opens the desired window.

1. Enter the following code for calling the full WebHelp help system for the web site or application immediately after the **heading** tag (<head>).

```
<html>
<head>
<script language="JavaScript">
    var strHelpOptions2 = "location=no";
    strHelpOptions2 += ",toolbar=yes";
    strHelpOptions2 += ",menubar=no";
    strHelpOptions2 += ",status=no";
    strHelpOptions2 += ",scrollbars=yes";
    strHelpOptions2 += ",resizable=yes";
    strHelpOptions2 += ",top=40";
    strHelpOptions2 += ",left=40";
    strHelpOptions2 += ",width=720";
    strHelpOptions2 += ",height=400";
```

2. Next, we'll enter a **ShowHelp2** function that opens a Window with the **strHelpOptions2** variable settings.

```
function ShowHelp2(strUrl)
{
    window.open(strUrl, "Help2", strHelpOptions2);
}
</script>
```

- Next, we need to update our Help button hyperlink to use the **JavaScript ShowHelp2** parameter, so the topic is displayed in another window.

Locate the following code.

```
<a href="Examples/Web_Help/helpfile.htm">
```

Next, add the text **javascript:ShowHelp2** in front of the html file, and a closing parenthesis and single quote at the end of the file, so that the hyperlink now appears like the following.

```
<a href='javascript:ShowHelp2("Examples/Web_Help/Helpfile.htm")'>
```

- Save the file by selecting **File – Save**.
- Go to **Windows Explorer** and double-click on the **Application.htm** file. Click on the help button after each field and view the help.

This completes the steps to call a WebHelp help file in a separate window.

In the next section, we will basically perform the exact same steps as we just completed, except that we will create another JavaScript variable and function to display the context-sensitive text into a smaller window with different options.

Displaying Context-Sensitive Help in a Separate Window

In this section, we will launch an individual HTML file when a button is selected by a field. To do this, we will perform basically the same steps as we did above in calling an entire help file.

First, we need to define the Window parameters for calling the html file. We will define a variable called **strHelpOptions** that merely defines the **window parameters**.

Next, we will use the JavaScript **window.open** function (called **ShowHelp**) that opens a window with the strHelpOptions variable settings.

Lastly, we will update our Question Mark button hyperlinks to call the JavaScript ShowHelp function, which opens the desired window.

To Call a Context-Sensitive Help Topic in a Separate Window

First, we need to define the Window parameters for calling the full Webhelp help system. We will define a variable called **strHelpOptions** that merely defines the **window parameters**.

- Copy the script for **strHelpOptions2** and paste **directly above** it. Then make the following changes, changing the variable to **strHelpOptions**, as shown below.

```
<html>
<head>
<script language="JavaScript">
    var strHelpOptions = "location=no";
    strHelpOptions += ",toolbar=no";
    strHelpOptions += ",menubar=no";
    strHelpOptions += ",status=no";
    strHelpOptions += ",scrollbars=no";
    strHelpOptions += ",resizable=yes";
    strHelpOptions += ",top=140";
    strHelpOptions += ",left=150";
    strHelpOptions += ",width=400";
    strHelpOptions += ",height=125";
```

- Next, we'll enter a **ShowHelp** function that opens a Window with the **strHelpOptions** variable settings. Again, you can copy the **existing ShowHelp2** function and paste below the code above.

```
function ShowHelp(strUrl)
{
    window.open(strUrl, "Help", strHelpOptions);
}
</script>
```

- Next, we need to update our Help button hyperlinks next to each field, and use the **JavaScript ShowHelp** parameter, so the topic is displayed in another window.

Locate the following code for **helpfile_cust_no.htm**.

```
<a href="Web_Help/helpfile_cust_no_.htm"
```

Next, add the text **'javascript:ShowHelp** in front of the html file, and a closing parenthesis and single quote at the end of the file, so that the hyperlink now appears like the following:

```
<a href='javascript:ShowHelp("Web_Help/helpfile_cust_no_.htm")'
```

- Locate the rest of the **HREF** links to html files for the **question mark.gif** file, and add the same **javascript:ShowHelp**(text as shown above. Be sure to add the closing) and single quote after the .htm" file.
- Save the file by selecting **File – Save**.
- Go to **Windows Explorer** and double-click on the **Application.htm** file.

Click on the help button after each field and view the help. A window similar to the following should display.

Home Products Services Downloads Contact Us Help

ASMO

WebSearch

Name

Enter the name of the company here. This field is 25 characters long.

Previous Next

Name: ?

Address 1: ?

Address 2: ?

City: ?

State: ?

Zip: ?

Submit Cancel

This Page Intentionally Left Blank

Index

A

About

- help files, 3
- map files, 42
- WebHelp, 45, 68

Adding

- a graphic, 33
- HTML topics, 29
- new topics, 25

AVI files

- inserting, 34

AVR

- assigning context-sensitive help, 59
- calling an HTML file, 54
- calling help from help menu, 53
- calling help using help button, 54
- creating context-sensitive help, 58
- enabling secondary windows, 61
- enabling WhatsThisHelp, 63
- specifying a help file, 58
- using the CommonDialog control, 57
- using the HelpContextID property, 59
- using the HelpKey property, 60
- using the WhatsThisHelp button, 63
- using the WhatsThisHelpID property, 63
- using Winhlp32 map parameter, 55
- using Winhlp32 topic ID parameter, 57
- Winhlp32 parameters, 57

C

Calling help

- from help menu, 53
- using HTML file, 54
- using CommonDialog control, 56
- using help button, 54

Calling web app

- from help button, 69
- in secondary window, 70, 71

Changing

- background color of popup, 37
- window settings, 35

Color of popup

- changing, 37

CommonDialog control

- using to call help, 57

Compiling

- help project, 31, 41

Components of a help file, 8

- Images, 10
- Index, 12
- Links, 11
- Macros, 13
- Table of Contents, 12
- Topics, 9
- types of links, 11
- Windows, 10

Connecting topics, 10

Consideration prior to creating help, 7

Contents tab

- creating, 32

Context-sensitive help

- assigning for AVR, 69
- creating, 27, 28, 58
- specifying a help project, 58

Creating

- contents tab, 32
- context-sensitive help, 28, 59
- JavaScript function, 70, 71
- JavaScript variable, 70, 71
- links, 26
- map file, 42, 43
- new help project?, 17
- new topics, 25

Cross-Platform help format, 5

- issues, 5
- strengths, 5

D

Displaying

- images, 10
- topics, 10
- windows, 10

E

Enabling

- secondary windows, 61
- Websearch, 40

Existing file

- inserting, 21



F

- Files
 - inserting multimedia, 34
- FrontPage 2000
 - using, 70

G

- Generating
 - WebHelp, 46
- Getting around in Robohelp, 19
- Graphic
 - adding, 33

H

- Help button
 - calling help, 54
- Help file
 - about, 3
 - adding a graphic, 33
 - adding HTML topics, 29
 - adding new topics, 25
 - calling from CommonDialog control, 56
 - calling from help button, 54
 - calling from help menu, 53
 - changing color of popup, 37
 - changing window settings, 35
 - compiling, 31, 41
 - components, 8
 - considerations prior to creating, 7
 - creating, 17
 - creating an index, 22
 - creating context-sensitive help, 27, 58
 - creating links, 26
 - creating map file, 42
 - creating table of contents, 32
 - enabling websearch, 40
 - how can you use?, 3
 - inserting multimedia files, 34
 - project settings tabs, 39
 - properties of a help topic, 23
 - saving, 22
 - setting project settings, 39
 - specifying for AVR app, 58
 - what is it?, 3
 - where do I start?, 7
 - which help authoring package?, 7
 - Winhlp32 parameters, 57
 - why create?, 3
- Help formats
 - Cross-Platforms, 5
 - HTML Help, 5
 - intranet, 6
 - type of, 4
 - Winhelp 3.1, 4

- WinHelp 95/98/NT, 4
- Help issues
 - Cross-Platform, 5
 - HTML Help, 5
 - Intranet, 6
 - WinHelp 3.1, 4
 - WinHelp 95/98/NT, 4
- Help menu
 - calling help from, 53
 - calling html file, 54
- Help project
 - compiling, 41
- Help strengths
 - Cross-Platform help, 5
 - HTML Help, 5
 - Intranet help, 6
 - WinHelp 95/98/NT, 4
- Help topic
 - properties of, 23
- HelpContextID property
 - using, 59
- HelpKey property
 - using, 60
- How can you use Help?, 3
- HTML file
 - calling from help menu, 54
- HTML help format?, 5
 - issues, 5
 - strengths, 5
- HTML topics
 - creating, 29

I

- I parameter, 57
- Images
 - adding, 33
 - displaying, 10
- Images and Multimedia, 10
- Index, 12
 - creating, 22
- Inserting
 - existing file, 21
 - multimedia files, 34
 - wave file, 34
- Intranet help format, 6
 - issues, 6
 - strengths, 6

J

- JavaScript function
 - creating, 70, 71
- JavaScript variable
 - creating, 70, 71
- javascript:ShowHelp, 72

L

- Links, 11
 - creating, 26
 - strategies for designing, 11
 - types of, 11
- Listing of
 - Winhlp32 parameters, 58

M

- Macros
 - changing color of popup, 37
 - overview, 13
- Manual conventions, 2
- Map file
 - creating, 42, 43
- Map number parameter
 - using, 55
- Multimedia files
 - inserting, 34

N

- n parameter, 55
- Navigating RoboHelp, 20
- New help file
 - creating, 17

O

- Overview
 - webhelp, 45, 68

P

- Popup color
 - changing, 37
- Project settings
 - setting, 39
 - specifying help file, 58
 - specifying secondary windows, 61
- Properties of a help topic, 23

R

- RoboHelp
 - adding a graphic, 33
 - adding HTML topics, 29
 - adding new topics, 25
 - changing popup color, 37
 - changing window settings, 35
 - compiling help project, 31, 41
 - creating a contents tab, 32
 - creating a map file, 43
 - creating an index, 22
 - creating context-sensitive help, 27
 - creating links, 26

- creating map file, 42
- enabling websearch, 40
- generating WebHelp, 46
- getting around in, 19
- inserting an existing file, 21
- inserting multimedia files, 34
- navigating, 20
- properties of a help topic, 23
- saving a file, 22, 23
- setting project settings, 39

S

- Saving
 - help file, 22
- Saving your work, 2
- Secondary windows
 - about, 35
 - enabling, 62
- Setting
 - project settings, 39
- Source files, 1
- Startup macro
 - changing color of popup, 37
- Step 1
 - time to complete, 15
 - what it will look like, 15
 - what we will cover, 15
 - what you will learn, 15
- Step 2
 - time to complete, 51
 - what it will look like, 51
 - what we will cover, 51
 - what you will learn, 51
- Step 3
 - time to complete, 67
 - what it will look like, 67
 - what we will cover, 67
 - what you will learn, 67
- Strategies for designing links, 11

T

- Table of contents, 12
- Time to complete
 - step 1, 15
 - step 2, 51
 - step 3, 67
- Topic ID parameter
 - using, 57
- Topic types, 9
- Topics, 9
 - connecting, 10
 - displaying, 10
- Tutorial
 - manual conventions, 2
 - saving your work, 2

- source files, 1
- using, 1
- Types of
 - links, 11
 - help formats, 4
 - topics, 9

U

Using

- HelpContextID property, 59
- HelpKey property, 59
- the tutorial, 1
- WhatsThisHelp button, 62
- WhatsThisHelpID property, 62

W

Wave file

- inserting, 34

Web app

- calling WebHelp from help button, 69
- calling WebHelp in a separate window, 70, 71

WebHelp

- calling from help button, 69
- calling in separate window, 70, 71
- generating, 46
- overview, 45, 68

Websearch

- enabling, 40

What Do you Want to Do?, 2

What is a Windows Help File?, 3

What is WebHel?, 68

What it will look like

- step 1, 15

- step 2, 51

- step 3, 67

What we will cover

- in Step 1, 16

- in step 2, 52

- in step 3, 68

What you will learn

- in step 1, 15

- in step 2, 51

- in step 3, 66

WhatsThisHelp

- enabling, 63

WhatsThisHelp button

- using, 62

WhatsThisHelpID property

- using, 62

Where do I start?, 7

Which help authoring package?, 7

Why create a Help File?, 3

Window settings

- changing, 35

Windows

- displaying, 10

WinHelp 3.1 help format

- issues, 4

WinHelp 95/98/NT help format

- issues, 4

- strengths, 4

Winhlp32 parameters

- I, 57

- n, 55

- listing of, 57

- using, 55, 57