Demo Help Contents INFO

The DEMO help file example makes use of many of the HelpGen <u>macros</u> and <u>RTF commands</u> that are documented with HelpGen.

- <u>Planning Your Help File</u>
- <u>Creating the Macro File</u>
- <u>Using HelpGen Macros and RTF Commands</u>
- <u>Creating the Project File</u>
- <u>Generating the RTF File</u>
- <u>Generating the Help File</u>
- <u>Testing the Help File</u>

INFO **Glossary**

If you are unfamiliar with how Windows Help works, choose **Help | How To Use Help** from the Help menu now.

Demo Information

This icon topic **MUST** be present.   It is activated by clicking on the icon in the Table of Contents.   You may place your own text here.

Planning Your Help File

Planning the layout and contents of the help file is probably the most important part of creating on-line help.   A poorly conceived and implemented help file is almost worthless to a user.   They may not be able to find anything they are looking for in the help file, and if they do find the correct item, it may not help with their problems.

This part of help file generation doesn't have much to do with the actual mechanics of creating the help file, but it **is** important, nevertheless. Some basic steps in planning a help file are detailed below.

- **Outline the Help Subjects**:   Plan to have a few major sections, and many minor sections.   For instance, one way to organize is to use the basic outline of your user manual.   Another way is shown below:

  Introduction
  How To
  Menu Items
      File
          Open
          Close
          Exit
      Edit
          Cut
          Copy
          Paste

  This outline can go down as many levels as you want, but anything more than about 4 levels is too complicated for a normal user to navigate.   Once you have this basic outline, you essentially have each of the topics that will be discussed in the help file.   Each level of the outline except the lowest level corresponds to a single display page in the help file.   Each topic on the lowest level of the outline corresponds to a single display page in the help file.

  Continuing with our example above, then, the Table of Contents page would contain the Introduction, How To and Menu Items topics.   If a user clicked on Menu Items, they would move to a page titled Menu Items, which contains File and Edit topics.   Clicking on the File item, they would move to a page containing the Open, Close and Exit topics.   And finally, clicking on Open would move to a page that explains the File | Open menu item.

- **Create Topic Tags**:   Each item on each level of the outline should have a single word 'tag' that can be used to reference it to.   Go through the outline and create a tag for each outline item.   We will use these when we add the topics to the macro file.

   Introduction - intro
   How To - howto
   Menu Items - menuitems
   　　　　File - file
   　　　　　　　　Open - fileopen
   　　　　　　　　Close - fileclose
   　　　　　　　　Exit - fileexit
   　　　Edit - edit
   　　　　　　　　Cut - editcut
   　　　　　　　　Copy - editcopy
   　　　　　　　　Paste - editpaste

- **Flesh Out the Outline**:   What you are really doing is writing a user manual that will be read on-line in a non-linear (hypertext) fashion. Much of what you would normally say in a manual will be included in the on-line help.

   Write explanations for each of the items on the lowest level of your outline.   Use graphics (.BMP files) and/or examples liberally to explain the subject (a picture really *is* worth 1000 words).

- **Mark Definitions**:   When you have completely written all the text for your help file, go back and circle the terms that you think the user will have trouble with, and that need to be defined.   These terms will form the basis of your glossary.   The first time such a term is encountered in *any* topic, it should be marked with a .p or .p1 and should be explained in a .pent entry.   Create a topic tag for each of the terms.

- **Create the Macro File**:   You are now ready to create and edit a macro file for your on-line help.

Creating the Macro File

Now that you have your help topics outlined and fleshed out, you can create and edit a macro file for your help.

1. Continuing with the example we started in **Planning Your Help File** let's create an initial macro file.   Click on **File | Open Macro File** and type in 'MYHELP' for a file name.   HelpGen will ask if you wish to create this file.   Click on the 'Yes' button and MYHELP will be created.

2. Now let's do some editing on this file.   Click on **Edit | Macro File.** The text editor should appear, with the newly created MYHELP.MAC displayed in it.   Let's use the DEMO.BMP file for the icon, change the table of contents header from xxxx to Table of Contents and add the first level of our outline to the table of contents topic:

```
.rem(===========================================================)
.rem(MYHELP.MAC)
.rem( )
.rem(Created by the HelpGen Help Generator, version 1.20)
.rem(Copyright © 1994 by Rimrock Software)
.rem(All rights reserved.)
.rem(===========================================================)

.start(main,Contents,DEMO.BMP,Table of Contents)
.top(Help)
.top(Table of Contents)

.s
.in
##.j(intro,Introduction)
##.j(howto,How To)
##.j(menuitems,Menu Items)
.un
.end

.ent(icon,Information,Icon )
This icon topic .b(MUST) be present.  It is activated by clicking on the
   icon in the Table of Contents.  You may place your own text here.
.end

.end_file
```

Note that we dress up the topics by indenting them (.in and .un) and by putting a bullet in front of them (##).   We shouldn't compile this until we add the entries for the three jumps we just added:

```
.start(main,Contents,DEMO.BMP,Table of Contents)
.top(Help)
.top(Table of Contents)
```

```
.s
.in
##.j(intro,Introduction)
##.j(howto,How To)
##.j(menuitems,Menu Items)
.un
.end

.ent(icon,Information,Icon )
This icon topic .b(MUST) be present.  It is activated by clicking on the
icon in the Table of Contents.  You may place your own text here.
.end

.ent(intro,Introduction, )
.end

.ent(howto,How To, )
.end

.ent(menuitems,Menu Items, )
.end

.end_file
```

Now we can compile it if we want to, but let's continue.   Next, we add the second level of the outline.   In this case, it means adding to the Menu Items topic, and adding new topics for the third outline level:

```
.ent(menuitems,Menu Items, )
.in
##.j(file,File)
##.j(edit,Edit)
.un
.end

.ent(file,File Menu,The )
.end

.ent(edit,Edit Menu,The )
.end

.end_file
```

We continue to add new jumps and new topics until we reach the last level of the outline, where all the text will be entered.

Using HelpGen Macros and RTF Commands

If you have looked at **Creating the Macro File** you will see that we have already used many of the basic HelpGen macros.   The remaining macros and RTF commands are used to primarily dress up the help file, to make it more interesting and easier to use for the user.

> For instance, we just used the \li and \ri RTF commands to squish this paragraph down.   This makes it stand out from surrounding text.

NOTE: Another way to make text stand out is to place it in a box (.box and .bend macros).   This is the *most* common method used in user manuals to highlight important information.   This particular box is a single line box with shading enabled.

Creating the Project File

When we are ready to compile our help file, we will need a project file to tell the compiler how to compile it.   Follow these steps when creating a project file:

1. Select the **Options | Project File** menu item.   Make sure that all the options listed here are set the way you want them.   Keep in mind that in the shareware version of HelpGen, you can't change the copyright notice here.

2. Select the **Build | Project File** menu item.   Answer 'Yes' to the question and a project file will be built for you.

Generating the RTF File

When you have finished editing the macro file, you will want to create an RTF file from the macro file.   Select the **Build | RTF File** menu item to perform this step.

Generating the Help File

When the RTF file has been built, HelpGen will enable the generation of a help file.   Select the **Build | HLP File** menu item and HelpGen will invoke the help compiler that you have told it to use (see **Options | Directories**).

Testing the Help File

After the help compiler generates a help file for you (assuming that it did so with no errors), you can test the file.   To do this, select the **Test** menu item.   HelpGen will invoke the WinHelp help engine, using your help file. You can then proceed with the testing.

Glossary

Macro
Project
RTF
RTF Commands

The replacement of a long series of <u>RTF</u> commands with a single command word and (optional) series of arguments.

A file with the extension .HPJ, that contains options that specifies how a help file is to be built.   Required by the help compiler.

Rich Text Format, a standard text file formatting standard. An RTF file consists of <u>RTF commands</u> and normal text.

Special commands in an <u>RTF</u> file that describe how text is to be displayed. These commands start with a backslash (\) character.