Help Macros

This document describes some plain and exotic uses of Help macros. It is provided as an aid in the use of Doctor Help. There are some small accompanying Help files to show some of the techniques. If you have some other cute applications of macros, please send me an example.

About this Help file

This Help file was created from a Word for Windows document using a macro written by Roger Hadgraft. Any reasonable Word file can be translated into a Windows Help file (with the aid of the Help Compiler of course). The macro is available as **drhelpe4.zip** on **ftp.cica.indiana.edu** (in subdirectory **pub/pc/win3/winword**, or possibly still in **pub/pc/win3/uploads**), and a registered version (US$20) is available by contacting **roger.hadgraft@eng.monash.edu.au**.

Click here for help in reading this Help file.

Overview

This file describes the use of the macros within the Help Compiler. It describes Inserted Macros (ie. those inserted in the text of your help file), Startup Macros (those that go in the Project File), and concludes with a List of Macros.

Inserted Macros

Inserted macros are those that run when the reader clicks on a word or graphic. In each case, there must be some text (or graphic) that is underlined (single or double) followed by a macro string which is hidden (Ctrl-H). The macro always starts with an exclamation mark: ! This is how it might appear on screen in Word:

Run Notepad!ExecProgram("notepad.exe", 0)

Execute a Program
Jump to another Help file
Jump to a Topic in Another File
Opening a second Help instance

Execute a Program

It is often useful to be able to launch a program from within a Help file. With this capability you could develop a specialised environment for running a small number of programs, or you might want to start a program within a tutorial document. The macro is:

    !ExecProgram("progname filename1 ...", n)

The n=0 for a Normal window, n=1 is minimised, and n=2 is maximised.

For instance, click below to run Notepad using: !ExecProgram("notepad.exe"):

    Run Notepad

Apart from running your normal garden variety applications, some interesting possibilities include adding sound effects to a button action or even running a piece of video (using Video for Windows). This would seem to be a terrific addition to on-line lecture notes.

Jump to another Help file

Sometimes you may have a collection of Help files which you want to link together into a Help system, or you might just have one other Help file (eg. DRHLPUSR.HLP) which you want to let the user read, without including it in your own file. You can allow the reader to jump to the contents page of that file with the JumpContents macro like this:

```
Jump to Outline Help!JumpContents("outline.hlp")
```

Jump to Outline Help

It is then possible to backtrack to the original file using the **Back** button. Also, if you click on the **History** button, you will see that the topics in the original file are available, eg.

Contents
WINHELP:Contents for How to Use Help
WINHELP:Copying a Help Topic onto the Clipboard
Contents

where the prefix WINHELP indicates a topic in that file.

Jump to a Topic in Another File

If you know a topic name in another Help file, you can jump to it with the JumpContext macro, eg:

```
Browse Buttons|JumpId("drhlpusr.hlp", "browseb")
```

Browse Buttons

This is handled by defining context names in the other file.
For some comments about backtracking from other files, see the topic: Jump to another Help file

Opening a second Help instance

Sometimes you may want to open another Help file in a second window, rather than replacing the one that is currently open (eg. Jump to another Help file or Jump to a Topic). You can do this using the ExecProgram macro by running another copy of winhelp.exe:

```
!ExecProgram("winhelp.exe drhlpusr.hlp")
```

See Dr Help User instructions

Don't forget to insert the winhelp.exe part. The reader of your document will need to close that window, or switch back to the original window by clicking on it. Which option you provide depends on the sophistication of your readers. Too many open windows on the screen can be confusing.

Startup Macros

You can do nifty things with macros by including them in your Project File. The default project file already contains some startup macros, eg. BrowseButtons() and some CreateButtons macros.

Initial Popup window
New Menu Item
Extra Buttons

Initial Popup window

This help file (POPUP.DOC) shows an initial popup that might, for instance, display a title and author and copyright.

Initial popup

New Menu Item

This help file (NEWMENU.DOC) adds an item to the Help menu which summons the Dr Help User file. This is now included in default.hpj.

New Menu item

Extra Buttons

Here is a Help file (BUTTONS.DOC) with extra buttons added. This might be suitable for inexperienced users.

Extra Buttons

List of Macros

Menu & Button Options

About()
Annotate()
Back()
BookmarkDefine()
BookmarkMore()
BrowseButtons()
Contents()
CopyDialog()
CopyTopic()
Exit()
FileOpen()
HelpOn()
History()
Next()
Prev()
Print()
PrinterSetup()
Search()

New Buttons

ChangeButtonBinding("button_id", "button_macro")
CreateButton("button_id", "name", "macro")
DestroyButton("button_id")
DisableButton("button_id")
EnableButton("button_id")

Adding to Menus

AddAccelerator(key, shift-state, "macro")
AppendItem("menu_id", "item_id", "item_name", "macro")
ChangeItemBinding("item_id", "item_macro")
DeleteItem("item_id")
DisableItem("item_id")
EnableItem("item_id")
InsertItem("menu_id", "item_id", "item_name", "macro", position)
InsertMenu("menu_id", "menu_name", menu_position)

Markers

DeleteMark("marker")
GotoMark("marker")
IfThen(IsMark("marker"),"macro")
IfThenElse(IsMark("marker"),"macro1", "macro2")
Not(IsMark("marker"))
SaveMark("marker")

Windows

CloseWindow("window_name")
FocusWindow("window_name")
PositionWindow(x, y, width, height, state, "window_name")

Programs

ExecProgram("command_line", display_state)

Jumps

JumpContents("filename")
JumpContext("filename", context_number)
JumpHelpOn()
JumpId("filename", "context_name")
JumpKeyword("filename", "keyword")

PopupContext("filename", context_number)
PopupId("filename", "context_name")

Calling DLLs

RegisterRoutine("DLL_name", "function_name", "format_spec")

Set

SetContents("filename", context_number)
SetHelpOnFile("filename")