*[1]#[2]$[3]+[4]K[5]**SHOWBIT** **included in** ***ShowBmp V 1.27***

high resolution bitmap and hotspot utility for the windows TM help system

Important note: Before trying this demo, please copy your bitmap test files 16.bmp, 256.bmp, 24.bmp, test.gif, test.pcx into the appropriate subdirectory of the showbit demo directory.

Show 16 color DIB bitmap
Show 256 color DIB bitmap
Show 24 Bit color DIB bitmap
New: Show 256 color GIF bitmap
New: Show PCX bitmap
Brandnew: Hotspots on high resolution images
Brandnew: Winhelp "on line" hotspot editor for legends, jumps, popups and sound replay
Combination of multiple graphs in one topic
Demonstration of MOVER command
Demonstration of MOVEL command
Demonstration of MOVET command
Demonstration of MOVEB command
Demonstration of STRETCH command
Combination of animated graphs in one topic

This is a software demo ("evaluation display"). Showbit.dll is part of ShowBmp V 1.27, which offers advanced MCI functions such as AVI, FLC, WAV play, CD AUDIO control, windows application control. Please register for the complete package via GO SWREG (#2365 and #2500).
Rudolf Bumm, Munich, Germany 100025,2206

1BuildAll
2Topic1
3Topic1
4index:00001
5Topic1;

*6#7$8+9K10MOVER Command

{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp MOVER 4}
This example shows a 256 color bitmap sliding in from the right side with speed 4. The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp MOVER 4}

Effect: Move DIB Bitmap from right side

Command: MOVER s where s determines the speed of movement

6BuildAll
7MOVER
8MOVER
9index:00002
10MOVER;

*11#12$13+14K15New Bitmap reference options

In general, it is possible to reference bitmaps and MCI files in a help project in two ways: First, you can specify the exact path in the ShoBmp command line.

{ewc showbit.dll,ShowBit_DLL,c:\bmp\256.bmp}

Second, you can specify a relative path referencing a subdirectory in the directory from which your help file is called in a ShowBmp command line:

{ewc showbit.dll,ShowBit_DLL,.\bmp\256.bmp}

Third, and as a new feature, you can enter all bitmaps and MCI files ***in the BAGGAGE section of your help file project and reference the file by a simple name***. This has great advantages if you want to distribute your help project (the bitmaps and MCI file will be invisible to your customers and will be included in the *.hlp file)

Example (part of your help project file (*.hpj)):

.
.

[CONFIG]
BrowseButtons()

[BUILDTAGS]
BuildAll

[BAGGAGE]
c:\mci\super.avi
c:\bmp\test.bmp
c:\dos\crazy.gif

[WINDOWS]
MAIN = ,,,,(192,192,192)

You can now call on every target system the file super.avi, test.bmp and crazy.gif by entering in your RTF help source code:

{ewc showbit.dll,ShowBit_DLL,crazy.gif}
{ewc showbit.dll,ShowBit_DLL,test.bmp}
{ewc showbit.dll,ShowBit_DLL,super.avi MCI_VIDEO}

11BuildAll
12reference
13reference
14index:000016
15Bitmap reference opetions;

*16#17$18+19K20MOVEL Command

{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp MOVEL 2}
This example shows a 256 color bitmap sliding in from the left side with speed 2. The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp MOVEL 2}

Effect: Move DIB Bitmap from left side

Command: MOVEL s where s determines the speed of movement

16BuildAll
17MOVEL
18MOVEL
19index:00003
20MOVEL;

*[21]#[22]$[23]+[24]K[25]MOVET Command

{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp MOVET 1}
This example shows a 256 color bitmap sliding in from the right side with speed 1. The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp MOVET 1}

Effect: Move DIB Bitmap from top down and stop when picture is in full sight

Command: MOVET s where s determines the speed of movement

21BuildAll
22MOVET
23MOVET
24index:00004
25MOVET;

*[26]#[27]$[28]+[29]K[30]MOVEB Command

{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp MOVEB 3}
This example shows a 256 color bitmap sliding in from the right side with speed 3. The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp MOVEB 3}

Effect: Move DIB Bitmap from bottom upwards until picture is in full sight

Command: MOVEB s where s determines the speed of movement

26BuildAll
27MOVEB
28MOVEB
29index:00005
30MOVEB;

*[31]#[32]$[33]+[34]K[35]STRETCH Command

{ewl showbit.dll,ShowBit_DLL,.\bmp\24.bmp STRETCH 1 40}
This example shows a 24 bit/pixel bitmap (DIB-format) growing from a predefined size in percent of the original picture until the full size is reached. The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\bmp\24.bmp STRETCH 1 40}

Effect: Stretch Bitmap from a predefined size until full size is reached

Command: STRECH s y where s determines the speed of movement and y determines the initial size of the bitmap in percent of the original size

31BuildAll
32STRETCH
33STRETCH
34index:00006
35STRETCH;

*[36]#[37]$[38]+[39]K[40]*Brandnew*: Winhelp "on line" hotspot editor

{ewc showbit.dll,ShowBit_DLL,.\hotspots\hs3.bmp}

Edit dialog box. Click on the rectangles for an explanation legend.

ShowBmp now supports free definition of hotspots on high resolution images., which is superior to the color and function limitation in SHG bitmaps. Definition of the hotspot areas and actions can be performed during a winhelp "on line" session, if you include the bitmap in your RTF source file by help of showbmp.dll. Your images are not altered by ShowBmp; in fact, the hotspot information is stored in a different file at the location of the primary image (extension *.HOT). This example shows a 256 color bitmap (GIF-format [*.gif]). The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\hotspot\hs3.gif HOTSPOT_EDIT}

If you have displayed your image on the help topic page, and provided you have specified HOTSPOT_EDIT, you can immediately use the right mouse button for definition of a "hotspot" (a rectangle on the bitmap). Watch Showbmp draw an appropriate rectangle. If you release the right mouse button, a dialog box (seen at the top) requests the hotspot text definition and the definition of the type of hotspot action.

Legend	***Hotspot text is displayed as a legend in a popup window***
Jump	***Hotspot text is taken as a context string for a hyperjump in the current help project***
Popup	***Hotspot text is taken as a context string for a popup link in the current help project***
Sound	***Hotspot text is taken as a path to a *.wav file for immediate replay***

Use the "Next" and "Prev" buttons for editing prior or following hotspots. Use "Delete" for abandoning the current hotspot. Use "Clear all" for deleting all hotspots of this image. Use "Ok" for acepting the current hotspot definition. Use "Visual display" to indicate that you want to display the hotspot rectangles to the user.

When you close the help topic window, the hotspot information is "intelligently" saved in a *.hot file with the name of the image at the path location of the image. The image stays unaltered.

Try this all out by using the image displayed below.

{ewc showbit.dll,ShowBit_DLL,.\hotspots\hs4.gif HOTSPOT_EDIT }

36BuildAll
37hotspotedit
38hotspotedit
39index:00008
40Hotspot Editor;

*41#42$43+44K45*Brandnew*: Hotspot display in high resolution bitmaps !

{ewl showbit.dll,ShowBit_DLL,.\hotspots\hs1.gif} ***Figure 1:*** **Image of a newly designed surgical instrument. Please touch the regions of interest for further explanation.**

ShowBmp now supports free definition of hotspots on high resolution images., which is superior to the color and function limitation in SHG bitmaps. Your images are not altered by ShowBmp; in fact, the hotspot information is stored in a different file at the location of the primary image (extension *.HOT). This example shows a color bitmap (GIF-format [*.gif]). The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\hotspot\hs1.gif}

If hotspots are found, they are displayed according to their definition when the user presses a single click on the hotspot surface. Four actions are provided as a reaction to hotspot activation:

Legend	***Hotspot text is displayed as a legend in a popup window***
Jump	***Hotspot text is taken as a context string for a hyperjump in the current help project***
Popup	***Hotspot text is taken as a context string for a popup link in the current help project***
Sound	***Hotspot text is taken as a path to a *.wav file for immediate replay***

Technical information

In this version, the number of hotspots per image is limited to 64, and the size of legend text is limited to 256 char per each hotspot. However, the structure of the *.HOT file permits future enhancements. The file structure is open. The information is stored in TAGS of 16 bytes each. A TAG has the following structure in C-notation:

```
#define L_POS 0x01
#define L_TEXT 0x02
#define L_TYPE 0x03
#define L_FIRST 0xfffe
#define L_LAST 0xffff

#define HOTSPOT_JUMP 1
#define HOTSPOT_LEGEND 2
#define HOTSPOT_POPUP 3
#define HOTSPOT_SOUND 4

structure tag
   {
   long type;
   long which;
```

41BuildAll
42hotspot
43hotspot
44index:00008
45Hotspots;

```
  long para1;
  long para2;
  }
```

The first TAG is always a TAG of the type L_START.

TYPE	L_START
WHICH	not used
PARA1	max. number of hotspots in this image
PARA2	not used

Then, for each hotspot ***at least*** the following three TAGS are defined and stored:

(1) L_POS

TYPE	L_POS
WHICH	running number of hotspot
PARA1	the high word identifies the x value of the left upper corner of the hotspot rectangle the low word identifies the y value of the upper left corner of the hotspot rectangle
PARA2	the high word identifies the x value of the lower right corner of the hotspot rectangle the low word identifies the y value of the lower right corner of the hotspot rectangle

TYPE	L_TYPE
WHICH	running number of hotspot
PARA1	Type Value of hotspot (see HOTSPOT defines ...)
PARA2	Visibility of rectangle (visible if > 0)

TYPE	L_TEXT
WHICH	running number of hotspot
PARA1	size of hotspot text in bytes
PARA2	offset of text in bytes from start of the file

The text information should follow immediately after the L_TEXT TAG.

At the end of a *.HOT file, the last TAG should have the type

TYPE	L_LAST
WHICH	not used
PARA1	not used
PARA2	not used

The file structure is a bit difficult though easy expandable; in case you all like it and - it will be possible to read the older *.HOT files even if we enhance the structure, because the number of TAGS per hotspot is not limited. For example, in the near future the HOT - file format will support polyline defined hotspots, circles and ellipses.

∗[46]#[47]$[48]+[49]K[50]

{ewl showbit.dll,ShowBit_DLL,.\hotspots\hs2.gif}
Figure 2: **Head of the newly designed instrument. Microscissors in working channel.**

46BuildAll
47hs2
48hs2
49index:00006
50hs2

51#52$53+54K55256 color GIF (.gif) bitmap display

{ewl showbit.dll,ShowBit_DLL,.\gif\test.gif}
This example shows your GIF color bitmap. Should display all resolutions. If not, please report. The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\gif\test.gif}

Effect: GIF color bitmap display

Command: no command. Specification of bitmap path mandatory.

Note: GIF color bitmaps can only be displayed correctly if the graphic extension of the computer supports the color resolution and if the adequate resolution is selected. Check the windows setup program in the main program group for this information.

51BuildAll
52gifbitmap
53gifbitmap
54index:00020
55RGB color GIF bitmap;

[56]#[57]$[58]+[59]K[60]PCX (.pcx) bitmap display

{ewl showbit.dll,ShowBit_DLL,.\pcx\test.pcx}
This example shows a PCX color bitmap. Should display all resolutions. The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\pcx\test.pcx}

Effect: PCX color bitmap display

Command: no command. Specification of bitmap path mandatory.

Note: PCX color bitmaps can only be displayed correctly if the graphic extension of the computer supports the color resolution and if the adequate resolution is selected. Check the windows setup program in the main program group for this information.

56BuildAll
57pcxbitmap
58pcxbitmap
59index:00020
60RGB color PCX bitmap;

61#62$63+64K65 16 color DIB (.bmp) bitmap display

{ewl showbit.dll,ShowBit_DLL,.\bmp\16.bmp}
This example shows a 16color bitmap (DIB-format [*.bmp]). The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\bmp\16.bmp}

Effect: 16 color bitmap display

Command: no command. Specification of bitmap path mandatory.

Note: 16 color bitmaps are displayed correctly on almost all color screens, so use them preferably.

61BuildAll
6216bitmap
6316bitmap
64index:00007
6516 color DIB bitmap;

66#67$68+69K70256 color DIB (.bmp) bitmap display

{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp HOTSPOT_EDIT}
This example shows a 256 color bitmap (DIB-format [*.bmp]). The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp}

Effect: 256 color bitmap display

Command: no command. Specification of bitmap path mandatory.

Note: 256 color bitmaps can only be displayed correctly if the graphic extension of the computer supports the 256 color resolution and if the adequate resolution is selected. Check the windows setup program in the main program group for this information.

66BuildAll
67256bitmap
68256bitmap
69index:00008
70256 color DIB bitmap;

[71]#[72]$[73]+[74]K[75]RGB (24 bit) color DIB (.bmp) bitmap display

{ewl showbit.dll,ShowBit_DLL,.\bmp\24.bmp}
This example shows a RGB color bitmap (DIB-format [*.bmp]). The appropriate call from a RTF file should be as follows:

Syntax:
{ewl showbit.dll,ShowBit_DLL,.\bmp\24.bmp}

Effect: RGB color bitmap display

Command: no command. Specification of bitmap path mandatory.

Note: RGB color bitmaps can only be displayed correctly if the graphic extension of the computer supports the color resolution and if the adequate resolution is selected. Check the windows setup program in the main program group for this information.

71BuildAll
72rgbbitmap
73rgbbitmap
74index:00008
75RGB color DIB bitmap;

*76#77$78+79K80Combination of multiple graphs in one help topic

{ewl showbit.dll,ShowBit_DLL,.\bmp\24.bmp}**Fig. 1**:16 M color bitmap

{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp}**Fig. 2**: 256 color bitmap

{ewl showbit.dll,ShowBit_DLL,.\bmp\16.bmp}**Fig. 3**: 16 color bitmap

Multiple bitmaps can be called on one single help topic. Please note, that on systems with color limited graphic extensions threre may be a slight interference between the colors of the bitmaps, which is in fact due to adaptation of the individual bitmap palette to the system palette. To overcome this, use bitmaps with identical color palettes or install a graphic card with increased number of colors to display.

76BuildAll
77multiple
78multiple
79index:00009
80Multiple bitmaps on one help topic;

*81#82$83+84K85Combination of multiple animated graphs in one help topic

{ewl showbit.dll,ShowBit_DLL,.\bmp\24.bmp STRETCH 30}**Fig. 1**:16 M color bitmap

{ewl showbit.dll,ShowBit_DLL,.\bmp\256.bmp MOVEL 1}**Fig. 2**: 256 color bitmap

{ewl showbit.dll,ShowBit_DLL,.\bmp\16.bmp MOVER 1}**Fig. 3**: 16 color bitmap

Multiple animated bitmaps can be called on one single help topic. Refer to the text in Combination of multiple graphs for further explanation. The author is aware of the fact that , dependent on the speed of the graphic card, there may be currently a inhomogeneous picture display if multiple moving images of different resolutions are to be displayed.

81BuildAll
82multipleani
83multipleani
84index:00009
85Multiple animated bitmaps on one help topic;