ImgLib Programmer's Reference

Welcome to ImgLib.   ImgLib allows you to incorporate support for many of the most popular raster image formats without having to write any custom image reading code.   ImgLib also works with **WinG**, if available, to speed up some of the operations associated with handling and displaying of raster images.   The following is a list of help topics available:

Overview
Library Requirements

ImgLib API Reference
Sample Programs

Credits and Copyright Information
Support
What's Next?

Shareware Notice
Registration Form
LZW Compression Notes

ImgLib Requirements

ImgLib is a Win32 DLL that has been tested and can be used in Windows NT, Windows 95, and Win32s.   As these systems are all different, ImgLib has different requirements running under each version of Windows.

The preferred way to develop applications with **ImgLib** is to use a 32-bit compiler capable of creating Win32 PE-executable files.   Using a 32-bit compiler will assure the best performance on any supported platform.   ImgLib DLL is in itself a Win32 PE-executable which means that to use it in 16-bit Windows 3.1, Win32s has to be present.   ImgLib was tested successfully on Win32s versions 1.15 and above.

Calling **ImgLib** from a 16-bit application using Universal Thunk (Win32s) or Generic Thunk (Windows NT or Windows95) is also supported.   16-bit applications should link with the 16-bit import library **ImgLib16.lib** that implements the thunking.   Under Win32s 2 other 32-bit files are required: **imglib32.dll** and **ilstub32.exe**.   These 2 executables are not required in Windows NT or Windows 95 as the 16-bit **imglib16.dll** is capable of calling the actual 32-bit routines in **ImgLib.dll** directly.   In either case, the developer using **ImgLib** never needs to be concerned about writing the thunking code.

WinG, although needed by some utility functions, is optional, but recommended.

ImgLib API

The following is a list of all currently available functions callable in ImgLib.

File I/O
    ReadFileIntoDIB
    ReadFileIntoDDB
    WriteDIBToFile

Image Processing
    BrightenDIB
    ClipDIB
    ExpandToTrueDIB
    GrayDIB
    HalftoneDIB
    MergeDIB
    MirrorDIB
    ReduceDIB
    RotateDIB
    SmoothDIB
    WinGHalftoneDIB
    ZoomDIB

Miscellaneous
    CopyDIB
    CreateDIBPalette
    DDBToDIB
    DIBFree
    DIBToDDB
    GetDIBSize
    GetImgLibVersion
    GetLastImgLibError

void DIBFree (pDIB)

LPVOID pDIB a pointer to a Device-Independent bitmap

The **DIBFree** function frees memory that was previously allocated to hold a devide-independent bitmap.

Returns:
This function does not return a value

Notes:
The **pDIB** parameter must have been previously returned by one of the ImgLib's functions like **ReadFileIntoDIB** or **WinGHalftoneDIB**.   Currently, device-independent bitmaps are allocated using **GlobalAlloc**, but the internal implementation of the functions that allocate memory for device-independent bitmaps may change in the future, so freeing it by doing **GlobalFree(GlobalHandle(pDIB))** may not work.   Not using **DIBFree** may be especially problematic when using **ImgLib** from 16-bit code.

See Also:
ReadFileIntoDIB, WinGHalftoneDIB

LPVOID ReadFileIntoDIB (pszFileName)

LPSTR pszFileName   a file name

The **ReadFileIntoDIB** function loads one of the supported file formats into memory as a device-independent bitmap. The bitmap is ready then to be displayed using one of the **GDI** or **WinG** functions.

Returns:
If successful, the return value is a pointer to a device-independent bitmap.   If unsuccessful, the function returns NULL. To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
As a convention, device-independent bitmaps are stored in memory as a complete **BITMAPINFO** structure followed immediately by the bitmap data.   This is the format expected by **WinG** or **DIB.DRV**.   Before displaying a device-independent bitmap on a palette-based display a GDI palette has to be created and selected into the display context.   This can be accomplished easily with a call to **CreateDIBPalette**.

See Also:
ReadFileIntoDDB, CreateDIBPalette

HBITMAP ReadFileIntoDDB (pszFName, pPal, bAnimPal)

LPSTR pszFName   file name
HPALETTE *pPal   pointer to store a created palette
BOOL bAnimPal   will the palette be used in palette animation?

The **ReadFileIntoDDB** function loads one of the supported file formats into memory as a device-dependent bitmap.   The bitmap is ready then to be displayed by using one of the GDI functions.

Returns:
If successful, the return value is a handle of a device-dependent bitmap.   If applicable, the location pointed to by **pPalette** will be filled with a palette handle corresponding to the bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
To get the size of the bitmap returned by this function, GDI's **GetObject** function should be used.   Once not needed, the bitmap can be freed by calling **DeleteObject**.   When dealing with true-color image files on 256-color displays, **ReadFileIntoDDB** tries to get the best performance vs. quality results.   This is done by performing a fast color quantization but no dithering.   Please note that **ReadFileIntoDDB** will usually still achieve better results than **WinGHalftoneDIB**.

See Also:
ReadFileIntoDIB, WinGHalftoneDIB

int GetLastImgLibError (void)

The GetLastImgLibError function returns the last error number that was generated by ImgLib.

Returns:
An integer corresponding to one of the predefined **ImgLib errors**.

Notes:
Most of the ImgLib functions can generate errors.   The error condition will usually be indicated by a NULL return value from one of the ImgLib API functions.   If multiple errors occur between calls to **GetLastImgLibError**, only the last error number will be returned, and the subsequent calls will return the same error number.   Below is a list of possible errors returned by **GetLastImgLibError:**

ERROR_READ_ACCESS_DENIED	Access to read the requested file has been denied
ERROR_WRITE_ACCESS_DENIED	Access to write the requested file has been denied
ERROR_NO_MEMORY	Out of memory processing the request
ERROR_NO_DLL	A DLL required for the requested operation was not found
ERROR_INVALID_POINTER	One of the pointer parameters or pointers used internally is not valid
ERROR_INVALID_ARGUMENT	One of the function arguments was not valid or out of range
ERROR_UNSUPPORTED_IMAGE	The image format attempted to read is not supported
ERROR_INCOMPATIBLE_IMAGE	The image format selected is incompatible with the image itself
ERROR_WRITE_ERROR	Error occurred while writing to file

Some situations may cause ImgLib to generate warnings, like extra bytes in scanline, short read, etc.   Those are generated for debugging purposes and are stored in a file called "ilerr.out".   The contents of this file is reset every time the library is initialized.

See Also:

LPVOID WinGHalftoneDIB (pDIB, phPal)

LPVOID pDIB   a 24-bit DIB pointer
HPALETTE *phPal   a pointer to a palette

The **WinGHalftoneDIB** dithers a 24-bit DIB using WinG's halftone functions, if available.

Returns:
If successful, the return value is a pointer to a device-independent bitmap.   If unsuccessful, the function returns NULL. To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
WinG must be present to invoke this call.   If it is not installed on the system, the function will do nothing and it will return NULL.   The returned device-independent bitmap must be freed by calling **DIBFree**.

See Also:
ReadFileIntoDIB, WinG, DIBFree

WinG

WinG (pronounced "Win Gee") is an optimized library designed to enable high-performance graphics techniques under Windows 3.x, Win32s, Windows NT 3.5, Windows 95, and future Windows releases. WinG allows the programmer to create a GDI-compatible HBITMAP with a Device Independent Bitmap (DIB) as the drawing surface. Programmers can use GDI or their own code to draw onto this bitmap, then use WinG to transfer it quickly to the screen. WinG also provides halftoning APIs that use the standard Microsoft halftone palette to support simulation of true color on palette devices.

All device independent bitmaps returned by ImgLib's functions are 100% compatible with all of WinG's functions.

hPal CreateDIBPalette (pDIB)

LPVOID pDIB   a device-independent bitmap pointer

The **CreateDIBPalette** function creates a palette suitable for displaying a given device-independent bitmap.

Returns:
If successful, the return value is a handle to a GDI palette.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
As a convention, device-independent bitmaps are stored in memory as a complete **BITMAPINFO** structure followed immediately by the bitmap data.   **CreateDIBPalette** relies on that fact and interprets the colors within the **BITMAPINFO** structure.

See Also:
ReadFileIntoDIB

HBITMAP DIBToDDB (pDIB, phPal)

LPVOID pDIB   a device-independent bitmap
HPALETTE phPal   a location to store a palette at

The **DIBToDDB** function converts from a device-independent to device-dependent bitmap format.

Returns:
If successful, the return value is a handle of a device-dependent bitmap.   If applicable, a palette suitable for displaying the bitmap is stored at the location pointed to by **phPal**.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
Device-independent bitmaps are not as easily manipulated as the device-dependent bitmaps when it comes to displaying, stretching, or cropping the images.   **WinG** tries to address some of the performance problems associated with device-independent bitmaps, but still many developers may be more comfortable with device-dependent bitmaps.

See Also:
ReadFileIntoDIB, CreateDIBPalette, WinG, DIBToDDB

LPVOID DDBToDIB (hBMP, hPal)

HBITMAP hBMP   a device-dependent bitmap
HPALETTE hPal   a GDI palette

The **DDBToDIB** function converts from a device-dependent to device-independent bitmap format.

Returns:
If successful, the return value is a pointer to a device-independent bitmap.   If unsuccessful, the function returns NULL. To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
The palette handle **hPal** is only required for palette-based bitmaps.   Device-dependent bitmaps are not suitable for saving to files and for use with **WinG**.   For these and other reasont it may be necessary to convert to device-independent format. Also, **WinG** allows direct modification of device-independent bitmap data which in certain cases can improve the performance of bitmap rendering by bypassing GDI.

See Also:
ReadFileIntoDIB, CreateDIBPalette, WinG, DIBToDDB

LPVOID BrightenDIB (pDIB, iPercent)

LPVOID pDIB   a device-independent bitmap
int iPercent   brightness percentage

The **BrightenDIB** function brightens or darkens a device-independent bitmap.

Returns:
If successful, the return value is a pointer to a new device-independent bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
The **iPercent** parameter specifies the brightness that the resulting bitmap will be as the percentage of the original bitmap. For example, using 110 as iPercent will increase the brightness of the resulting bitmap by 10 percent, and using 90 will darken the returned bitmap by 10 percent.   Please note that palette-based device-independent bitmaps can be processed very quickly because only the color table needs to be modified, e.g. a 256-color bitmap will require 256 modifications regardless of the image size.   True color bitmaps, on the other hand require that each pixel value be modified, so an 800x600 pixel image will require 400000 color modifications.   Also, brightening the image, and then immeditely darkening it by the same percentage will not always result in an image that's identical to the original.   Calculation roundoffs as well as the maximum and minimum brightness levels are the cause of that behavior.   For example, since the maximum color brightness can be 255, brightening white (255,255,255) by 10% will result in the same color, but darkening it by the same amount will result in (229,229,229).

See Also:
ReadFileIntoDIB

LPVOID ReduceDIB (pDIB, lColors, bDither)

LPVOID pDIB   a device-independent bitmap
long lColors   maximum of colors in the resulting bitmap
BOOL bDither   dither the image using the Floyd-Steinberg dithering

The **ReduceDIB** function performs color reduction on a device-independent bitmap.   The returned device-independent bitmap will always be palette-based and because of this restriction the maximum value of lColors can be 256.

Returns:
If successful, the return value is a pointer to a device-independent bitmap.   If unsuccessful, the function returns NULL. To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
The **lColors** parameter specifies the maximum number of colors that the returned bitmap will contain.   Note that the returned bitmap may contain fewer colors than **lColors**.   This occurs most often on gray-scale and palette-based images and is a result of the color reduction algorithm creating duplicate color entries which then are eliminated.   Color reduction of true color images has fewer chances in causing this fenomenon because of the vast number of unique colors in typical true color images.

See Also:
ExpandToTrueDIB, ReadFileIntoDIB, DIBFree

1

ImgLib General Overview

ImgLib is a 32-bit DLL that implements raster image reading, converting, and manipulating functionalities required by today's multimedia applications.   ImgLib is being offered as shareware to offer a competitive alternative to many expensive commercial packages.   ImgLib's attractive pricing is also possible because portions of its code have been adopted from freely available X windows sources.

For full credits see the **Credits** section elsewhere in this document.   Besides the ability to read the most popular raster image formats ImgLib offers added functionality to modify the bitmaps in memory to aid in displaying them on all existing Windows-capable displays with best possible results and without sacrificing the image quality.

The most current version of ImgLib is available at:
ftp://ftp.netcom.com/pub/si/simsoft/ImgLib/ImgLibXX.zip
 or
ftp://ftp.cica.indiana.edu/pub/pc/win3/programr/imglibXX.zip
 (where XX indicates the current version of ImgLib)

The following is a list of image formats read by ImgLib:

Windows and OS/2 BMP, JPEG, PC Paintbrush (PCX), TIFF, Targa, Photo CD, MacPaint, FBM, Sun Raster, CMU WM Raster, Portable Bit Map (PBM, PGM, PPM), Faces Project, Utah RLE, X Window Dump, McIDAS areafile, G3 FAX, GEM Bit, X Pixmap, and X Bitmap.

LPVOID SmoothDIB (pDIB, iIterations)

LPVOID pDIB   a device-independent bitmap
short iIterations   number of times that the smoothing should be applied

The **SmoothDIB** function performs smoothing on a device-independent bitmap.   To some, 'bluring' would be a more accurate term to describe this effect.

Returns:
If successful, the return value is a pointer to another device-independent bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
The smoothing algorithm is quite effective at removing lossy image compression's artifacts.   It may also effectively clean up images that have been poorly dithered from true color to 256-color.   Unfortunately, smoothing reaults in some loss of image detail, so applying it more than once is not advised unless an "out-of-focus" effect is desired.   The returned device-independent bitmap is always in true-color 24-bit format.

See Also:
ReduceDIB, **ReadFileIntoDIB**, **DIBFree**

LPVOID HalftoneDIB (pDIB)

LPVOID pDIB   a device-independent bitmap

The **HalftoneDIB** function performs black-and-white halftonning on a device-independent bitmap.

Returns:
If successful, the return value is a pointer to a device-independent monochrome bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
The **HalftoneDIB** function performs color-to-monochrome halftonning that makes the resulting monochrome bitmap suitable for printing on a black-and-white printer.   Although Windows' **StretchDIBits** is still the recommended way of getting bitmaps printed, this function may sometimes achieve more desirable results.

See Also:
ReadFileIntoDIB, **DIBFree**

LPVOID GrayDIB (pDIB)

LPVOID pDIB   a device-independent bitmap

The **GrayDIB** function converts a color device-independent bitmap to grayscale.

Returns:
If successful, the return value is a pointer to a new grayscale device-independent bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
The **GrayDIB** function performs grayscaling based on the color intensity of each pixel.   Please note that palette-based device-independent bitmaps can be processed very quickly because only the color table needs to be modified, e.g. a 256-color bitmap will require 256 modifications regardless of the image size.   True color bitmaps, on the other hand require that each pixel value be modified, so an 800x600 pixel image will require 400000 color modifications.

See Also:
ReadFileIntoDIB, DIBFree

Credits

All ImgLib original code, this helpfile, and the sample programs are distributed under the following Copyright:

Copyright (c) 1994, 1995, SimSoft

Portions of the ImgLib code are based on XLI, an X Windows-based image viewing utility.   XLI is copywritten material with a very loose copyright allowing unlimited modification and distribution if the copyright notices are left intact. Various portions are copywritten by various people, but all use a modification of the MIT copyright notice.   The intent is to keep the source free, not to stifle its distribution, so please write to **SimSoft** or **Graeme Gill** if you have any questions. The original sources for XLI are available at: ftp://ftp.x.org/contrib/applications/xli.1.16.tar.gz

Here are the names of the main contributors to XLI and their copyright stamements:

Copyright (c) 1989, 1990, 1991, 1992, 1993 Jim Frost, Graeme Gill and others
Copyright (c) 1991 Tim Northrup
Copyright (c) 1990 Mark Majhor
Copyright (c) 1989, 1990 Kirk L. Johnson

Here is the XLI permission stamement.   It also applies to the **sample code only** distributed with Imglib:

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. The author makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

This helpfile was developed using Help Pre-Compiler v3.0 by Antonio Cordero Balcazar.   The newest version of Help Pre-Compiler is available at ftp://ftp.cica.indiana.edu/pub/pc/win3/util.   The author can be reached at L0063@albeniz.eui.upm.es.   Help Pre-Compiler is freeware.

TIFF file support comes from a library written by Sam Leffler.
Here is the TIFF library copyright and permission statement:

Tag Image File Format Library

Copyright (c) 1988, 1989, 1990, 1991, 1992, 1993, 1994 Sam Leffler
Copyright (c) 1991, 1992, 1993, 1994 Silicon Graphics, Inc.

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notices and this permission notice appear in all copies of the software and related documentation, and (ii) the names of Sam Leffler and Silicon Graphics may not be used in any advertising or publicity relating to the software without the specific, prior written permission of Sam Leffler and Silicon Graphics.

THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Only the registered users of ImgLib have the right to redistribute the following files free of any royalties:

> ImgLib.dll
> imglib16.dll
> imglib32.dll
> ilstub32.exe

The following applies to all of ImgLib package:

SimSoft

SimSoft specializes in Windows and Win32-based software development.   Custom programming and consulting is available.   For more information contact us at:

SimSoft
P.O. Box 4091
Redondo Beach, CA 90278
USA

or by e-mail at:

arybicki@netcom.com

Graeme Gill is the maintainer of Xli.   He can be reached via email as: graeme@labtam.oz.au

Sample Programs

Currently, there is only one sample program, but it uses most of the ImgLib's functionality.   The main file is tstapp.c and it can be compiled as either 16 or 32-bit program.   The corresponding tstapp.h and tstapp.rc can also be compiled under either 16 or 32-bit environment.   There is no need for the module definition file in the 32-bit version, but the 16-bit version should use tstapp16.def.

16-bit programs should link with imglib16.lib which will bind them to imglib16.dll which in turn will communicate with the main imglib.dll.   32-bit programs can link directly with imglib.lib which will bind them to the main imglib.dll.

Initially, a C++ program was developed to serve as a test application, but it was dependent on MFC 3.0 which is not available to everyone.   For that reason, a plain C program (tstapp.c) was developed sacrificing some of the nice user-interface features in favor of portability and readability.   This way one file (tstapp.c) contains all the sample code in contrast to many files generated by the Visual C++ environment.

Please remember that if you distribute your programs as 32-bit applications you only need to include imglib.dll in addition to your program files.   When distributing a 16-bit program, however, you will need to include imglib.dll, imglib16.dll, imglib32.dll, and ilstub32.exe.   The last 2 of those are required for Win32s support and would not be necessary for Windows 95 or NT.

What's Next?

Features planned for future releases:

Writing DIBs in more than BMP and TIFF file formats.
Add a function to identify an image without loading it.
Allow to supress the warnings or to place them in a file different than "ilerr.out".
Add short read error notification.
Add support for 16 and 32-bit DIB handling.
Expose the ImgLib functionality via an OLE 2.0 automation server.
Additional image processing functions (your suggestions are welcome, e-mail **the author**)

ImgLib Bug Report
=================
mail to arybicki@netcom.com

Your e-mail address:

Is the problem repeatable?
 Always[ ] Sometimes[ ] Happened only once[ ]

Which most closely describes the problem?
 Access violation[ ] Nothing returned[ ] Corrupt image[ ]
 System hung[ ] Corrupt file[ ] Documentation error[ ]

Suspected area of problem:
 ImgLib[ ] Universal Thunk code[ ] Generic Thunk code[ ]

Is ImgLib useable after this problem?
 Yes[ ] No[ ]

Is there a particular image file that exposes this problem?
 Yes[ ] No[ ]
    If yes, describe how I can ftp it for testing.

Describe the problem and steps to take to repeat the problem here:

Which version of Windows do you develop under?
 Win31[ ] Windows 95[ ] Windows NT[ ]

What type of programs do you use ImgLib with?
 16-bit[ ] 32-bit[ ]

Describe the compiler/development environment details here:

How much RAM do you have?

Enter any suggestions for improving ImgLib here:

BOOL WriteDIBToFile (pDIB, pszFileName)

LPVOID pDIB   a device-independent bitmap
LPSTR pszFileName   file name
short nFileType   type of the file to write the DIB as

The **WriteDIBToFile** function saves a given devide-independent bitmap in a Windows bitmap (.BMP) or TIFF file.   See the **Notes**section for more details.

Returns:
If successful, the return value is TRUE.   If unsuccessful, the function returns FALSE.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
All device-independent bitmaps are supported ranging from 1-bit to 24-bit.   The following table lists all currently supported file types for saving.

FILETYPE_BMP	Windows uncompressed BMP file
FILETYPE_TIFF_DEFAULT	TIFF file. Let ImgLib pick the compression depending on DIB type.
FILETYPE_TIFF_NO_COMPRESSION	Uncompressed TIFF file.
FILETYPE_TIFF_HUFFMAN	Modified Huffman compressed TIFF. Monochrome DIBs only.
FILETYPE_TIFF_PACKBITS	Run-length encoded TIFF.
FILETYPE_TIFF_G3	CCITT Group 3 encoded TIFF. Monochrome DIBs only.
FILETYPE_TIFF_G4	CCITT Group 4 encoded TIFF. Monochrome DIBs only.

See Also:
ReadFileIntoDIB, **DIBFree**

LPVOID MergeDIB (pDIB1, pDIB2, lX, lY)

LPVOID pDIB1   a device-independent bitmap
LPVOID pDIB2   bitmap to merge onto pDIB1
long lX   x offset at which to place pDIB2 onto pDIB1
long lY   y offset at which to place pDIB2 onto pDIB1

The **MergeDIB** function merges a device-independent bitmap onto another (usually larger) device-independent bitmap. Merging is performed in true color mode to avoid any color information loss.   This function is very useful when creating a collection of small images as one larger picture.

Returns:
If successful, the return value is a pointer to a new device-independent bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
If pDIB2 extends outside of lDIB1, it will be clipped.   The resulting device-independent bitmap will have the same dimensions as pDIB1.   In 256-color mode, **ReduceDIB** can be used to get a device-independent bitmap from the true color image returned by **MergeDIB**.   As always, the returned device-independent bitmap should be freed with a call to **DIBFree**.

See Also:
ReduceDIB, DIBFree

Technical Support

The unlimited technical support for registered users will be provided via E-mail by **SimSoft**.

The most current version of ImgLib will always be available at ftp://ftp.netcom.com/pub/si/simsoft/ImgLib/ImgLibXX.zip or ftp://ftp.cica.indiana.edu/pub/msdos/win3/programr/imglibXX.zip (where XX indicates the current version of ImgLib).

Please report all bugs by using the included **bug report form**.   To save the form, just copy it to the clipboard from this helpfile, and then paste it to your editor and save.   Try to include as much information as you can.   If you find a picture file that should be but is not read properly by ImgLib, we can arrange for you to be able to forward it to me, but I will not make my anonymous directory write-enabled unless it is for a short period of time and I am logged in at the same time.

Unless someone's life depends on it, SimSoft will not remove any features from **ImgLib**, so you may depend on their presence in the future releases.   For the list of features that I am planning to add to ImgLib, see the **What's Next?** page.

LPVOID RotateDIB (pDIB, iDegrees)

LPVOID pDIB   a device-independent bitmap
short iDegrees   degree of rotation (90, 180, 270)

The **RotateDIB** function rotates a device-independent bitmap clockwise by the requested angle.

Returns:
If successful, the return value is a pointer to a new device-independent bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:

See Also:
MirrorDIB, DIBFree

LPVOID MirrorDIB (pDIB, bVertical)

LPVOID pDIB   a device-independent bitmap
BOOL bVertical   mirror vertically if this parameter is nonzero

The **MirrorDIB** function creates a mirror image of a device-independent bitmap.

Returns:
If successful, the return value is a pointer to a new device-independent bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:

See Also:
RotateDIB, DIBFree

LPVOID ExpandToTrueDIB (pDIB)

LPVOID pDIB   a device-independent bitmap

The **ExpandToTrueDIB** function converts a device-independent bitmap to true-color 24-bit device-independent format.

Returns:
If successful, the return value is a pointer to a new device-independent bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
This function basically does the opposite of **ReduceDIB**, but without any information loss.

See Also:
ReduceDIB, **DIBFree**

BOOL GetImgLibVersion (pszBuffer, iLen)

LPSTR pszBuffer a pointer to receive the ImgLib version information string
short iLen   length of the buffer pointed to by pszBuffer

The **GetImgLibVersion** function returns a string containing ImgLib's version as well as the copyright and registration information.

Returns:
TRUE if successfull, otherwise FALSE.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
Depending on the size of the ImgLib registration information the length of the buffer required to hold all the information will vary.   In most cases a 400-character buffer will suffice.   The returned string will bot be NULL-terminated if pszBuffer was not sufficiently large to hold all the information.

See Also:

Shareware Notice

ImgLib is distributed as shareware.   It may be evaluated without charge for 30 days, and after that period it should be registered.   Please note that the unregistered version of **ImgLib** may not be distributed with your software.

Registration entitles you to free technical support via e-mail, free bugfix releases, and advance information about new features and new products from **SimSoft**.

Most importantly, the registered users receive a fully optimized version of **ImgLib**.   Our benchmarks show an average 33% performance improvement in the registered version over the shareware version.

Once the registration fee has been received, **SimSoft** will send the registered version of **ImgLib** to you electronically. This can be done either via ftp (you name the site, and we'll put it there) or via e-mail with UUENCODE or MIME. We welcome your suggestions for other efficient methods of delivering **ImgLib** to you electronically.

The US $50.00 registration fee can be paid by mail to **SimSoft** by filling out the **Registration Form** and attaching a check or money order.

Site and source code licensing is available but the terms will be negotiated on individual basis.   Contact **SimSoft** for details.

ImgLib Registration Form (complete all fields)
=======================

Your e-mail address:

Your postal address:

Which version of Windows do you develop under?
 Win31[ ] Win95[ ] WinNT[ ]

What type of programs do you use ImgLib with?
 16-bit[ ] 32-bit[ ]

How did you obtain ImgLib?
 ftp[ ] Friend[ ] BBS[ ] Other[ ] (describe below)

Include a check or money order for US $50.00 and mail this form to:

SimSoft
P.O. Box 4091
Redondo Beach, CA 90278
USA

Enter any suggestions for improving or adding to ImgLib below.

LZW Compression Notes

When GIF and TIFF with LZW image file formats were implemented, the LZW algorithm was thought to be in public domain.   This, apparently, is not true.   Unis Corporation is the owner of United States patent number 4,558,302 and corresponding foreign patents.   For a long time, Unisys Corporation did nothing to enforce the patent, but in 1994 it came to an agreement with Compuserve, the copyright holder of GIF specification, that GIF developers have to license LZW from Unisys.

Unfortunately, SimSoft was unable to obtain a fair license from Unisys while still maintaining its commitment to distribute ImgLib royalty-free.   For that reason, all code that uses the LZW algorithm was removed from ImgLib.   This means that ImgLib will not longer be able to support reading GIF files and those of TIFF files that saved using the LZW compression. As of TIFF specification version 6, the LZW compression is optional and should not be used to assure compatibility with "baseline" TIFF readers.   SimSoft will search for public domain or freeware libraries capable of reading GIF, and if we locate such a tool we will happy to recommend it.

SimSoft apologizes for the inconvinience that this may cause you, but we had no control over the issue.   Please contact **SimSoft** if you have suggestions on this subject.

LPVOID ClipDIB (pDIB, lX, lY, lWidth, lHeight)

LPVOID pDIB   a device-independent bitmap
long lX   x coordinate of the upper-left corner
long lY   y coordinate of the upper-left corner
long lWidth   width to clip
long lHeight   height to clip

The **ClipDIB** function creates a new device-independent bitmap by clipping a section of existing device-independent bitmap.

Returns:
If successful, the return value is a pointer to a new device-independent bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
This function is very useful when implementing copy/cut/paste clipboard operations.   See the sample program for details.
As always, the returned device-independent bitmap must be released by calling **DIBFree**.

See Also:
ReadFileIntoDIB

LPVOID ZoomDIB (pDIB, lXPercent, lYPercent)

LPVOID pDIB   a device-independent bitmap
long lXPercent x zoom scale
long lYPercent y zoom scale

The **ZoomDIB** function creates a new device-independent bitmap by scaling existing device-independent bitmap by a given percentage.

Returns:
If successful, the return value is a pointer to a new device-independent bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
A scaling parameter of 0 or 100 is interpretted as no scaling in that direction.   If no scaling is requested in both directions, the source bitmap is returned.   As always, the returned device-independent bitmap must be released by calling **DIBFree**.

See Also:
ReadFileIntoDIB

LPVOID CopyDIB (pDIB)

LPVOID pDIB   a device-independent bitmap

The **CopyDIB** function creates a new device-independent which is the same as the original.

Returns:
If successful, the return value is a pointer to a new device-independent bitmap.   If unsuccessful, the function returns NULL.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
As always, the returned device-independent bitmap must be released by calling **DIBFree**.

See Also:
ReadFileIntoDIB, **GetDIBSize**

DWORD GetDIBSize (pDIB)

LPVOID pDIB   a device-independent bitmap

The **GetDIBSize** function returns the total amount of memory occupied by the given device-independent bitmap.

Returns:
If successful, the return value is total size of the specified device-independent bitmap.   If unsuccessful, the function returns 0.   To get a specific reason for the failure, call **GetLastImgLibError**.

Notes:
This function may be useful if the device-independent bitmap needs to be copied into a buffer that cannot be managed by ImgLib.   See the sample program for details of how to use this function in order to be able to interact with Windows clipboard.

See Also:
ReadFileIntoDIB