

# License

Installing and using this library (**GFL SDK/GfIAX**) signifies acceptance of these terms and conditions of the license.

- "**GFL SDK/GFLAx**" is provided as **Freeware** for private non-commercial or educational use (including non-profit organization).

You must contact me for commercial use and distribution.

[webmaster@xnview.com](mailto:webmaster@xnview.com)

- You may not use GFL SDK or GfIAX to create components or controls to be used by other developers without written approval.
- The product developed by the Licensee should not be similar to or should not compete with XnView/NConvert (should not be a graphic viewer or converter).
- "GFL SDK/GFLAx" IS NOT DESIGNED, MANUFACTURED OR INTENDED FOR USE OR RESALE AS ONLINE CONTROL EQUIPMENT IN HAZARDOUS ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS IN THE OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION OR COMMUNICATIONS SYSTEMS, AIR TRAFFIC CONTROL, DIRECT LIFE SUPPORT MACHINES, OR WEAPONS SYSTEMS, IN WHICH THE FAILURE OF "GFL SDK/GFLAx" COULD LEAD DIRECTLY TO DEATH, PERSONAL INJURY, OR SEVERE PHYSICAL OR ENVIRONMENTAL DAMAGE.
- "GFL SDK/GFLAx" 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 the author of this software be held liable for

data loss, damages, loss of profits or any other kind of loss while using or misusing this software.

- You may not use, copy, emulate, clone, rent, lease, sell, modify, decompile, disassemble, otherwise reverse engineer, or transfer the licensed program, or any subset of the licensed program, except as provided for in this agreement. Any such unauthorized use shall result in immediate and automatic termination of this license and may result in criminal and/or civil prosecution.

**Important** The use of LZW technology needs to be licensed separately from UNISYS Corporation. Contact UNISYS to get this license ([www.unisys.com](http://www.unisys.com)).

For JPEG-2000 & JBIG use, see corresponding licenses in Plugins folder of the GFL SDK package

Any suggestions, feedback and comments are welcome.

# gflLibraryInit

The **gflLibraryInit** function initialize the library. Must be used before call of GFL's functions.

```
GFL_ERROR gflLibraryInit(  
    void  
);
```

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflLibraryInitEx](#), [gflLibraryExit](#), [gflEnableLZW](#), [gflGetVersion](#), [gflGetErrorString](#)

# gflLibraryInitEx

The **gflLibraryInitEx** function initialize the library. Must be used before call of GFL's functions.

```
GFL_ERROR gflLibraryInitEx(  
    GFL_ALLOC_CALLBACK alloc_callback,  
    GFL_REALLOC_CALLBACK realloc_callback,  
    GFL_FREE_CALLBACK free_callback,  
    void * user_parms  
);
```

## Parameters

*alloc\_callback*

Pointer to an alloc user function. (void \* (GFLAPI \*)(  
 GFL\_UINT32 size, void \* user\_parms ))

*realloc\_callback*

Pointer to a read user function. (void \* (GFLAPI \*)(  
 void \* ptr, GFL\_UINT32 new\_size, void \* user\_parms ))

*free\_callback*

Pointer to a read user function. (void (GFLAPI \*)(  
 void \* ptr, void \* user\_parms ))

*user\_parms*

User parameter used in callback.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflLibraryInit](#), [gflLibraryExit](#), [gflEnableLZW](#), [gflGetVersion](#), [gflGetErrorString](#)

# gflLibraryExit

The **gflLibraryExit** function frees the library.

```
void gflLibraryExit(  
    void  
);
```

## See also

[gflLibraryInit](#), [gflLibraryInitEx](#), [gflEnableLZW](#), [gflGetVersion](#)

# gflEnableLZW

The **gflEnableLZW** function enables the use of LZW compression. **Be careful!** there is a patent from **Unisys**.

```
void gflEnableLZW(  
    GFL_BOOL enable  
);
```

## Parameters

enable

GFL\_TRUE or GFL\_FALSE.

By default, the LZW compression is not enabled.

## See also

[gflLibraryInit](#), [gflLibraryExit](#), [gflGetVersion](#)

# gflGetVersion

The **gflGetVersion** function returns the GFL's version.

```
const char * gflGetVersion(  
    void  
);
```

## Return value

Pointer to a null-terminated string that contains version of GFL.

## See also

[gflLibraryInit](#), [gflLibraryExit](#), [gflEnableLZW](#)

# gflSetPluginsPathname

The **gflSetPluginsPathname** function allows to set a Plugin's folder. Be careful, this function must be called before `gflLibraryInit`.

```
void gflSetPluginsPathname(  
    const char * pathname  
);
```

## Parameters

`pathname`

Pointer to a null-terminated string that contains the pathname of plugins.

## See also

[gflLibraryInit](#), [gflLibraryExit](#), [gflEnableLZW](#)

# gflAllocBitmap

The **gflAllocBitmap** function allocates a picture, and return a pointer on a [GFL\\_BITMAP](#) structure.

```
GFL_BITMAP * gflAllocBitmap(  
    GFL_BITMAP_TYPE bitmap_type,  
    GFL_INT32 width,  
    GFL_INT32 height,  
    GFL_UINT32 line_padding,  
    const GFL_COLOR * color  
);
```

## Parameters

*bitmap\_type*

Type of picture wanted.

GFL_BINARY	0x0001	Binary (8 bits)
GFL_GREY	0x0002	Grey scale (8 bits)
GFL_COLORS	0x0004	Colors with colormap (8 bits)
GFL_RGB	0x0008	TrueColors - Red/Green/Blue (24 bits)
GFL_RGBA	0x0010	TrueColors - Red/Green/Blue/Alpha (32 bits)
GFL_BGR	0x0020	TrueColors - Blue/Green/Red (24 bits)
GFL_ABGR	0x0040	TrueColors - Alpha/Blue/Green/Red (32 bits)
GFL_BGRA	0x0100	TrueColors - Blue/Green/Red/Alpha (32 bits)
GFL_ARGB	0x0200	TrueColors - Alpha/Red/Green/Blue (32 bits)

*width*

Width of the picture wanted.

*height*

Height of the picture wanted.

line\_padding

Pad for a pixel line.

For a value of 4, each line of pixels have a multiple of 4 bytes (32 bits).

color

Pointer to a [GFL\\_COLOR](#) structure used to set the background color.

Can be NULL, the background color is (0,0,0).

## Return value

A pointer to a [GFL\\_BITMAP](#) structure or NULL.

## See also

[gflFreeBitmap](#), [gflFreeBitmapData](#)

# gflAllocBitmapEx

The **gflAllocBitmapEx** function allocates a picture, and return a pointer on a [GFL\\_BITMAP](#) structure.

```
GFL_BITMAP * gflAllocBitmapEx(  
    GFL_BITMAP_TYPE bitmap_type,  
    GFL_INT32 width,  
    GFL_INT32 height,  
    GFL_UINT16 bits_per_component,  
    GFL_UINT32 line_padding,  
    const GFL_COLOR * color  
);
```

## Parameters

*bitmap\_type*

Type of picture wanted.

GFL_BINARY	0x0001	Binary (8 bits)
GFL_GREY	0x0002	Grey scale (8 bits)
GFL_COLORS	0x0004	Colors with colormap (8 bits)
GFL_RGB	0x0008	TrueColors - Red/Green/Blue (24 bits)
GFL_RGBA	0x0010	TrueColors - Red/Green/Blue/Alpha (32 bits)
GFL_BGR	0x0020	TrueColors - Blue/Green/Red (24 bits)
GFL_ABGR	0x0040	TrueColors - Alpha/Blue/Green/Red (32 bits)
GFL_BGRA	0x0100	TrueColors - Blue/Green/Red/Alpha (32 bits)
GFL_ARGB	0x0200	TrueColors - Alpha/Red/Green/Blue (32 bits)

*width*

Width of the picture wanted.

*height*

Height of the picture wanted.

bits\_per\_component  
Bits per component wanted. Can be 8 or 16.

line\_padding  
Pad for a pixel line.  
For a value of 4, each line of pixels have a multiple of 4 bytes (32 bits).

color  
Pointer to a [GFL\\_COLOR](#) structure used to set the background color.  
Can be NULL, the background color is (0,0,0).

## Return value

A pointer to a [GFL\\_BITMAP](#) structure or NULL.

## See also

[gflFreeBitmap](#), [gflFreeBitmapData](#)

# gflFreeBitmap

The **gflFreeBitmap** function frees a [GFL\\_BITMAP](#) structure, and his content.

```
void gflFreeBitmap(  
    GFL_BITMAP * bitmap  
);
```

## Parameters

*bitmap*

Pointer to a [GFL\\_BITMAP](#) structure.

## See also

[gflAllocBitmap](#), [gflFreeBitmapData](#)

# gflCloneBitmap

The **gflCloneBitmap** function allows to clone a bitmap, and returns a pointer to a [GFL\\_BITMAP](#) structure.

```
GFL_BITMAP * gflCloneBitmap(  
    const GFL_BITMAP * bitmap  
);
```

## Parameters

*bitmap*  
Pointer to a [GFL\\_BITMAP](#) structure.

## Return value

A pointer to a [GFL\\_BITMAP](#) structure or NULL.

## See also

[gflAllocBitmap](#), [gflFreeBitmap](#), [gflFreeBitmapData](#)

# gflFreeBitmapData

The **gflFreeBitmapData** function frees the content of a [GFL\\_BITMAP](#) structure, and his content.

```
void gflFreeBitmapData(  
    GFL_BITMAP * bitmap  
);
```

## Parameters

*bitmap*  
Pointer to a [GFL\\_BITMAP](#) structure.

## See also

[gflAllocBitmap](#), [gflFreeBitmap](#)

# gflMemoryAlloc

The **gflMemoryAlloc** function allocates memory.

```
void * gflMemoryAlloc(  
    GFL_UINT32 size  
);
```

## Parameters

size  
Size wanted.

## Return value

The function returns NULL if an error occurs.

## See also

[gflMemoryRealloc](#), [gflMemoryFree](#)

# gflMemoryRealloc

The **gflMemoryRealloc** function reallocates a memory area.

```
void * gflMemoryRealloc(  
    void * ptr,  
    GFL_UINT32 size  
);
```

## Parameters

ptr  
 Pointer to a memory area allocated.

size  
 New size.

## Return value

The function returns NULL if an error occurs.

## See also

[gflMemoryAlloc](#), [gflMemoryFree](#)

# gflMemoryFree

The **gflMemoryFree** function frees memory.

```
void gflMemoryFree(  
    void * ptr  
);
```

## Parameters

ptr  
 Pointer to a memory area allocated.

## See also

[gflMemoryAlloc](#), [gflMemoryRealloc](#)

# gflGetNumberOfFormat

The **gflGetNumberOfFormat** function gets the number of formats available in GFL.

```
GFL_INT32 gflGetNumberOfFormat(  
    void  
);
```

## Return value

Number of formats.

## See also

[gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#), [gflFormatsWritableByIndex](#),  
[gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflFormatsReadableByName](#), [gflGetDefaultFormatSuffixByIndex](#),  
[gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#), [gflGetFormatDescriptionByName](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflFormatIsSupported

The **gflFormatIsSupported** function determines if a format is available in GFL.

```
GFL_BOOL gflFormatIsSupported(  
    const char * name  
);
```

## Parameters

*name*

Pointer to a null-terminated string that contains the name of the format.

## Return value

GFL\_FALSE or GFL\_TRUE.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsWritableByIndex](#), [gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflFormatsReadableByName](#), [gflGetDefaultFormatSuffixByIndex](#), [gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#), [gflGetFormatDescriptionByName](#), [gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflGetFormatNameByIndex

The **gflGetFormatNameByIndex** function returns name of a format's index.

```
const char * gflGetFormatNameByIndex(  
    GFL_INT32 index  
);
```

## Parameters

*index*  
Index of the format.

## Return value

Pointer to a null-terminated string that contains name of the format.  
NULL if there is an error.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflFormatsSupported](#), [gflFormatsWritableByIndex](#),  
[gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflFormatsReadableByName](#), [gflGetDefaultFormatSuffixByIndex](#),  
[gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#), [gflGetFormatDescriptionByName](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflGetFormatIndexByName

The **gflGetFormatIndexByName** function returns index of a format's name.

```
GFL_INT32 gflGetFormatIndexByName(  
    const char * name  
);
```

## Parameters

name

Pointer to a null-terminated string that contains the name of the format.

## Return value

Pointer to a null-terminated string that contains the name of the format.

NULL if there is an error.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatNameByIndex](#), [gflFormatIsSupported](#), [gflFormatIsWritableByIndex](#),  
[gflFormatIsWritableByName](#), [gflFormatIsReadableByIndex](#), [gflFormatIsReadableByName](#), [gflGetDefaultFormatSuffixByIndex](#),  
[gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#), [gflGetFormatDescriptionByName](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflFormatIsReadableByIndex

The **gflFormatIsReadableByIndex** function determines if a format is readable with its index.

```
GFL_BOOL gflFormatIsReadableByIndex(  
    GFL_INT32 index  
);
```

## Parameters

*index*  
Index of format.

## Return value

GFL\_FALSE or GFL\_TRUE.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),  
[gflFormatsWritableByIndex](#), [gflFormatsWritableByName](#), [gflFormatsReadableByName](#), [gflGetDefaultFormatSuffixByIndex](#),  
[gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#), [gflGetFormatDescriptionByName](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflFormatIsReadableByName

The **gflFormatIsReadableByName** function determines if a format is readable with its name.

```
GFL_BOOL gflFormatIsReadableByName(  
    const char * name  
);
```

## Parameters

name

Pointer to a null-terminated string that contains the name of the format.

## Return value

GFL\_FALSE or GFL\_TRUE.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),  
[gflFormatsWritableByIndex](#), [gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflGetDefaultFormatSuffixByIndex](#),  
[gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#), [gflGetFormatDescriptionByName](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflFormatIsWritableByIndex

The **gflFormatIsWritableByIndex** function determines if a format is writable with its index.

```
GFL_BOOL gflFormatIsWritableByIndex(  
    GFL_INT32 index  
);
```

## Parameters

*index*  
Index of the format.

## Return value

GFL\_FALSE or GFL\_TRUE.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),  
[gflFormatIsWritableByName](#), [gflFormatIsReadableByIndex](#), [gflFormatIsReadableByName](#), [gflGetDefaultFormatSuffixByIndex](#),  
[gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#), [gflGetFormatDescriptionByName](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflFormatIsWritableByName

The **gflFormatIsWritableByName** function determines if a format is writable with its name.

```
GFL_BOOL gflFormatIsWritableByName(  
    const char * name  
);
```

## Parameters

*name*

Pointer to a null-terminated string that contains the name of the format.

## Return value

GFL\_FALSE or GFL\_TRUE.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),  
[gflFormatIsWritableByIndex](#), [gflFormatIsReadableByIndex](#), [gflFormatIsReadableByName](#), [gflGetDefaultFormatSuffixByIndex](#),  
[gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#), [gflGetFormatDescriptionByName](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflGetFormatDescriptionByIndex

The **gflGetFormatDescriptionByIndex** function returns the label of a format's index.

```
const char * gflGetFormatDescriptionByIndex(  
    GFL_INT32 index  
);
```

## Parameters

*index*  
Index of the format.

## Return value

Pointer to a null-terminated string that contains the label.  
NULL if there is an error.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),  
[gflFormatsWritableByIndex](#), [gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflFormatsReadableByName](#),  
[gflGetDefaultFormatSuffixByIndex](#), [gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByName](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflGetFormatDescriptionByName

The **gflGetFormatDescriptionByName** function returns the label of a format's name.

```
const char * gflGetFormatDescriptionByName(  
    const char * name  
);
```

## Parameters

name

Pointer to a null-terminated string that contains the name of the format.

## Return value

Pointer to a null-terminated string that contains the label. NULL if there is an error.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),  
[gflFormatsWritableByIndex](#), [gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflFormatsReadableByName](#),  
[gflGetDefaultFormatSuffixByIndex](#), [gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflGetDefaultFormatSuffixByIndex

The **gflGetDefaultFormatSuffixByIndex** function returns the default extension of a format's index.

```
const char * gflGetDefaultFormatSuffixByIndex(  
    GFL_INT32 index  
);
```

## Parameters

*index*  
Index of the format.

## Return value

Pointer to a null-terminated string that contains the default extension.  
NULL if there is an error.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),  
[gflFormatsWritableByIndex](#), [gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflFormatsReadableByName](#),  
[gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#), [gflGetFormatDescriptionByName](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflGetDefaultFormatSuffixByName

The **gflGetDefaultFormatSuffixByName** function returns the default extension of a format's name.

```
const char * gflGetDefaultFormatSuffixByName(  
    const char * name  
);
```

## Parameters

name

Pointer to a null-terminated string that contains the name of the format.

## Return value

Pointer to a null-terminated string that contains the default extension.

NULL if there is an error.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),  
[gflFormatsWritableByIndex](#), [gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflFormatsReadableByName](#),  
[gflGetDefaultFormatSuffixByIndex](#), [gflGetFormatDescriptionByIndex](#), [gflGetFormatDescriptionByName](#),  
[gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflGetFormatInformationByIndex

The **gflGetFormatInformationByIndex** function retrieves all informations of a format's index.

```
GFL_ERROR gflGetFormatInformationByIndex(  
    GFL_INT32 index,  
  
    GFL_FORMAT_INFORMATION * informations  
);
```

## Parameters

*index*

Index of the format.

*informations*

Pointer to a [GFL\\_FORMAT\\_INFORMATION](#) structure.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),  
[gflFormatsWritableByIndex](#), [gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflFormatsReadableByName](#),  
[gflGetDefaultFormatSuffixByName](#), [gflGetDefaultFormatSuffixByIndex](#), [gflGetFormatDescriptionByIndex](#),  
[gflGetFormatDescriptionByName](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflGetFormatInformationByName

The **gflGetFormatInformationByName** function retrieves all informations of a format's name.

```
GFL_ERROR gflGetFormatInformationByName(  
    const char * name,  
  
    GFL_FORMAT_INFORMATION * informations  
);
```

## Parameters

name

Pointer to a null-terminated string that contains the name of the format.

informations

Pointer to a [GFL\\_FORMAT\\_INFORMATION](#) structure.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),  
[gflFormatsWritableByIndex](#), [gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflFormatsReadableByName](#),  
[gflGetDefaultFormatSuffixByName](#), [gflGetDefaultFormatSuffixByIndex](#), [gflGetFormatDescriptionByIndex](#),  
[gflGetFormatDescriptionByName](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#), [GFL\\_SAVE\\_PARAMS](#)

# gflGetDefaultLoadParams

The **gflGetDefaultLoadParams** function sets the [GFL\\_LOAD\\_PARAMS](#) structure with default values. To use before call of [gflLoadBitmap](#).

```
void gflGetDefaultLoadParams(  
    GFL_LOAD_PARAMS * load_params  
);
```

## Parameters

load\_params

Pointer to a [GFL\\_LOAD\\_PARAMS](#) structure.

## See also

[gflGetDefaultThumbnailParams](#), [gflGetDefaultSaveParams](#), [gflLoadBitmap](#), [gflLoadBitmapFromHandle](#),

# gflLoadBitmap

The **gflLoadBitmap** function load a picture file into memory.

```
GFL_ERROR gflLoadBitmap(  
    const char * filename,  
    GFL_BITMAP ** bitmap,  
    GFL_LOAD_PARAMS * params,  
    GFL_FILE_INFORMATION * informations,  
);
```

## Parameters

filename

Pointer to a null-terminated string that contains the filename to load.

bitmap

Address of a pointer to a [GFL\\_BITMAP](#) structure.

params

Pointer to a [GFL\\_LOAD\\_PARAMS](#) structure.

This structure must be filled correctly.

informations

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure. Can be NULL if you don't want it.

You must use [gflFreeInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflGetDefaultSaveParams](#), [gflLoadBitmapFromMemory](#),  
[gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromMemory](#), [gflLoadThumbnailFromHandle](#), [gflSaveBitmap](#),  
[gflSaveBitmapIntoMemory](#), [gflSaveBitmapIntoHandle](#)

# gflLoadBitmapFromMemory

The **gflLoadBitmapFromMemory** function load a picture from memory.

```
GFL_ERROR gflLoadBitmapFromMemory(  
    GFL_UINT8 * data,  
    GFL_UINT32 data_length,  
    GFL_BITMAP ** bitmap,  
    GFL_LOAD_PARAMS * params,  
    GFL_FILE_INFORMATION * informations,  
);
```

## Parameters

data

Pointer to the picture.

data\_length

Length of data.

bitmap

Address of a pointer to a [GFL\\_BITMAP](#) structure.

params

Pointer to a [GFL\\_LOAD\\_PARAMS](#) structure.

This structure must be filled correctly.

informations

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure. Can be NULL if you don't want it.

You must use [gflFreeInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflGetDefaultSaveParams](#), [gflLoadBitmap](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromMemory](#), [gflLoadThumbnailFromHandle](#), [gflSaveBitmap](#), [gflSaveBitmapIntoMemory](#), [gflSaveBitmapIntoHandle](#)

# gflLoadBitmapFromHandle

The **gflLoadBitmapFromHandle** function load a picture into memory with the use of read callback functions.

```
GFL_ERROR gflLoadBitmapFromHandle(  
    GFL_HANDLE handle,  
    GFL_BITMAP ** bitmap,  
    GFL_LOAD_PARAMS * params,  
    GFL_FILE_INFORMATION * informations,  
);
```

## Parameters

handle

User handle. The Callbacks field of the [GFL\\_LOAD\\_PARAMS](#) structure must be filled correctly.

bitmap

Address of a pointer to a [GFL\\_BITMAP](#) structure.

params

Pointer to a [GFL\\_LOAD\\_PARAMS](#) structure.

This structure must be filled correctly.

informations

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure. Can be NULL if you don't want it.

You must use [gflFreeInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflGetDefaultSaveParams](#), [gflLoadBitmap](#),

[gflLoadBitmapFromMemory](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromMemory](#), [gflLoadThumbnailFromHandle](#), [gflSaveBitmap](#),

[gflSaveBitmapIntoMemory](#), [gflSaveBitmapIntoHandle](#)

# gflGetDefaultPreviewParams

The **gflGetDefaultPreviewParams** function sets the [GFL\\_LOAD\\_PARAMS](#) structure with default values. To use before call of [gflLoadPreview](#).

```
void gflGetDefaultPreviewParams(  
    GFL_LOAD_PARAMS * load_params  
);
```

## Parameters

load\_params

Pointer to a [GFL\\_LOAD\\_PARAMS](#) structure.

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultSaveParams](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromHandle](#),

# gflLoadThumbnail

The **gflLoadThumbnail** function load a picture file as a thumbnail into memory.

```
GFL_ERROR gflLoadThumbnail(  
    const char * filename,  
    GFL_INT32 width,  
    GFL_INT32 height,  
    GFL_BITMAP ** bitmap,  
    GFL_LOAD_PARAMS * params,  
    GFL_FILE_INFORMATION * informations,  
);
```

## Parameters

filename

Pointer to a null-terminated string that contains the filename to load.

width

Width of the thumbnail.

height

Height of the thumbnail.

bitmap

Address of a pointer to a [GFL\\_BITMAP](#) structure.

params

Pointer to a [GFL\\_LOAD\\_PARAMS](#) structure.

This structure must be filled correctly.

informations

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure. Can be NULL if you don't want it.

You must use [gflFreeInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflGetDefaultSaveParams](#), [gflLoadBitmap](#),  
[gflLoadBitmapFromMemory](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnailFromMemory](#), [gflLoadThumbnailFromHandle](#),  
[gflSaveBitmap](#), [gflSaveBitmapIntoMemory](#), [gflSaveBitmapIntoHandle](#)

# gflLoadThumbnailFromMemory

The **gflLoadThumbnailFromMemory** function load a picture file as a thumbnail from memory.

```
GFL_ERROR gflLoadThumbnailFromMemory(  
    GFL_UINT8 * data,  
    GFL_UINT32 data_length,  
    GFL_INT32 width,  
    GFL_INT32 height,  
    GFL_BITMAP ** bitmap,  
    GFL_LOAD_PARAMS * params,  
    GFL_FILE_INFORMATION * informations,  
);
```

## Parameters

data

Pointer to the picture.

data\_length

Length of data.

width

Width of the thumbnail.

height

Height of the thumbnail.

bitmap

Address of a pointer to a [GFL\\_BITMAP](#) structure.

params

Pointer to a [GFL\\_LOAD\\_PARAMS](#) structure.

This structure must be filled correctly.

informations

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure. Can be NULL if you don't want it.

You must use [gflFreeFileInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a

value of [GFL\\_ERROR](#).

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflGetDefaultSaveParams](#), [gflLoadBitmap](#),  
[gflLoadBitmapFromMemory](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromHandle](#), [gflSaveBitmap](#),  
[gflSaveBitmapIntoMemory](#), [gflSaveBitmapIntoHandle](#)

# gflLoadThumbnailFromHandle

The **gflLoadThumbnailFromHandle** function load a picture file as a thumbnail into memory with the use of read callback functions.

```
GFL_ERROR gflLoadThumbnailFromHandle(  
    GFL_HANDLE handle,  
    GFL_INT32 width,  
    GFL_INT32 height,  
    GFL_BITMAP ** bitmap,  
    GFL_LOAD_PARAMS * params,  
    GFL_FILE_INFORMATION * informations,  
);
```

## Parameters

handle

User handle. The Callbacks field of the [GFL\\_LOAD\\_PARAMS](#) structure must be filled correctly.

width

Width of the thumbnail.

height

Height of the thumbnail.

bitmap

Address of a pointer to a [GFL\\_BITMAP](#) structure.

params

Pointer to a [GFL\\_LOAD\\_PARAMS](#) structure.

This structure must be filled correctly.

informations

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure. Can be NULL if you don't want it.

You must use [gflFreeFileInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflGetDefaultSaveParams](#), [gflLoadBitmap](#),  
[gflLoadBitmapFromMemory](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromMemory](#), [gflSaveBitmap](#),  
[gflSaveBitmapIntoMemory](#), [gflSaveBitmapIntoHandle](#)

# gflGetFileInformation

The **gflGetFileInformation** function retrieves all informations about a picture file.

```
GFL_ERROR gflGetFileInformation(  
    const char * filename,  
  
    GFL_INT32 index,  
  
    GFL_FILE_INFORMATION * information  
);
```

## Parameters

filename

Pointer to a null-terminated string that contains the filename.

index

Index of format. -1 for automatic recognition.

information

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure.

You must use [gflFreeFileInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflLoadBitmap](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromHandle](#)

# gflGetFileInformationFromMemory

The **gflGetFileInformationFromMemory** function retrieves all informations about a picture from memory.

```
GFL_ERROR gflGetFileInformationFromMemory(  
    GFL_UINT8 * data,  
    GFL_UINT32 data_length,  
  
    GFL_INT32 index,  
  
    GFL_FILE_INFORMATION * information  
);
```

## Parameters

data

Pointer to the picture.

data\_length

Length of data.

index

Index of format. -1 for automatic recognition.

information

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure.

You must use [gflFreeFileInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflLoadBitmap](#), [gflLoadBitmapFromMemory](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromMemory](#), [gflLoadThumbnailFromHandle](#)

# gflGetFileInformationFromHandle

The **gflGetFileInformationFromHandle** function retrieves all informations about a picture with the use of read callback functions.

```
GFL_ERROR gflGetFileInformationFromHandle(  
    GFL_HANDLE handle,  
  
    GFL_INT32 index,  
  
    const GFL_LOAD_CALLBACKS * callbacks,  
  
    GFL_FILE_INFORMATION * information  
);
```

## Parameters

handle

User handle.

index

Index of format. -1 for automatic recognition.

callbacks

Callback to access picture data.

information

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure.

You must use [gflFreeFileInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflLoadBitmap](#), [gflLoadBitmapFromMemory](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromMemory](#), [gflLoadThumbnailFromHandle](#)

# gflFreeFileInformation

The **gflFreeFileInformation** function frees the content of a [GFL\\_FILE\\_INFORMATION](#) structure.

```
void gflFreeFileInformation(  
    GFL_FILE_INFORMATION * information  
);
```

## Parameters

bitmap

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure.

## See also

[gflGetFileInformation](#), [gflLoadBitmap](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromHandle](#)

# gflGetDefaultSaveParams

The **gflGetDefaultSaveParams** function sets the [GFL\\_SAVE\\_PARAMS](#) structure with default values. To use before call of [gflSaveBitmap](#).

```
void gflGetDefaultSaveParams(  
    GFL_SAVE_PARAMS * save_params  
);
```

## Parameters

save\_params

Pointer to a [GFL\\_SAVE\\_PARAMS](#) structure.

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflSaveBitmap](#), [gflSaveBitmapIntoHandle](#)

# gflSaveBitmap

The **gflSaveBitmap** function save a picture in memory into a file.

```
GFL_ERROR gflSaveBitmap(  
    char * filename,  
    const GFL_BITMAP * bitmap,  
    GFL_SAVE_PARAMS * params,  
);
```

## Parameters

filename

Pointer to a null-terminated string that contains the filename to save.

bitmap

Pointer to a [GFL\\_BITMAP](#) structure.

params

Pointer to a [GFL\\_SAVE\\_PARAMS](#) structure.

This structure must be filled correctly, in particular the FormatIndex field.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflGetDefaultSaveParams](#), [gflLoadBitmap](#),  
[gflLoadBitmapFromMemory](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromMemory](#),  
[gflLoadThumbnailFromHandle](#), [gflSaveBitmapIntoMemory](#), [gflSaveBitmapIntoHandle](#)

# gflSaveBitmapIntoHandle

The **gflSaveBitmapIntoHandle** function save a picture in memory into a file with the use of write callback functions.

```
GFL_ERROR gflSaveBitmapIntoHandle(  
    GFL_HANDLE handle,  
    const GFL_BITMAP * bitmap,  
    GFL_SAVE_PARAMS * params,  
);
```

## Parameters

handle

User handle. The Callbacks field of the [GFL\\_SAVE\\_PARAMS](#) structure must be filled correctly.

bitmap

Address of a pointer to a [GFL\\_BITMAP](#) structure.

params

Pointer to a [GFL\\_SAVE\\_PARAMS](#) structure.

This structure must be filled correctly, in particular the FormatIndex field.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflGetDefaultSaveParams](#), [gflLoadBitmap](#),  
[gflLoadBitmapFromMemory](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromMemory](#),  
[gflLoadThumbnailFromHandle](#), [gflSaveBitmapIntoMemory](#), [gflSaveBitmap](#)

# gflFileCreate

The **gflFileCreate** function creates a multi-page file.

```
GFL_ERROR gflFileCreate(  
    GFL_FILE_HANDLE * handle,  
    const char * filename,  
    GFL_UINT32 image_count,  
    GFL_SAVE_PARAMS * params  
);
```

## Parameters

handle

Address of an handle.

filename

Pointer to a null-terminated string that contains the filename to create.

image\_count

Number of picture to be added.

params

Pointer to a [GFL\\_SAVE\\_PARAMS](#) structure.  
This structure must be filled correctly.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflFileAddPicture](#), [gflFileClose](#)

# gflFileAddPicture

The **gflFileAddPicture** function add a picture to a multi-page file.

```
GFL_ERROR gflFileAddPicture(  
    GFL_FILE_HANDLE handle,  
    const GFL_BITMAP * bitmap  
);
```

## Parameters

handle

Handle of the file.

bitmap

Pointer to a [GFL\\_BITMAP](#) structure. This is the picture to add.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflFileCreate](#), [gflFileClose](#)

# gflFileClose

The **gflFileClose** function closes a multi-page file.

```
void gflFileClose(  
    GFL_FILE_HANDLE handle  
);
```

## Parameters

*handle*  
Handle of file.

## See also

[gflFileCreate](#), [gflFileAddPicture](#)

# gflGetErrorString

The **gflGetErrorString** function returns a null-terminated string that contains the error string.

```
const char * gflGetErrorString(  
    GFL_ERROR error  
);
```

## Parameters

error

### Erreur possible

GFL_NO_ERROR	0	No error
GFL_ERROR_FILE_OPEN	1	File open error
GFL_ERROR_FILE_READ	2	File read error
GFL_ERROR_FILE_CREATE	3	File create error
GFL_ERROR_FILE_WRITE	4	File write error
GFL_ERROR_NO_MEMORY	5	No more memory
GFL_ERROR_UNKNOWN_FORMAT	6	Unknown format
GFL_ERROR_BAD_BITMAP	7	The format doesn't permit to save this type of picture
GFL_ERROR_BAD_FORMAT_INDEX	10	Bad picture format
GFL_ERROR_BAD_PARAMETERS	50	Bas parameters
GFL_UNKNOWN_ERROR	255	Other error

## Return value

The function returns a null-terminated string that contains the error string.

# gflBitmapSetComment

The **gflBitmapSetComment** function change the comment associated with a bitmap. Only some formats can save the comment.

```
void gflSetComment(  
    GFL_BITMAP * bitmap,  
  
    const char * comment  
);
```

## Parameters

*bitmap*

Pointer to a [GFL\\_BITMAP](#) structure.

*comment*

Pointer to a null-terminated string that contains the comment.

## See also

[gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#),

[gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflFreeIPTC](#)

# gflBitmapHasEXIF

The **gflBitmapHasEXIF** function is used to know if the picture has EXIF metadata.

```
GFL_BOOL gflBitmapHasEXIF(  
    GFL_BITMAP * bitmap  
);
```

## Parameters

*bitmap*

Pointer to a [GFL\\_BITMAP](#) structure.

## Return value

The function returns GFL\_TRUE if the bitmap has EXIF metadata.

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#),  
[gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflBitmapSetIPTC](#), [gflFreeIPTC](#), [gflNewIPTC](#), [gflSetIPTCValue](#), [gflRemoveIPTCValue](#),  
[gflLoadIPTC](#), [gflSaveIPTC](#)

# gflBitmapRemoveEXIFThumbnail

The **gflBitmapRemoveEXIFThumbnail** function remove thumbnail from EXIF metadata.

```
GFL_ERROR gflBitmapRemoveEXIFThumbnail(  
    GFL_BITMAP * bitmap  
);
```

## Parameters

*bitmap*

Pointer to a [GFL\\_BITMAP](#) structure.

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#), [gflBitmapHasIPTC](#),  
[gflBitmapGetIPTC](#), [gflFreeIPTC](#)

# gflBitmapGetEXIF

The **gflBitmapGetEXIF** function returns EXIF metadata in a readable form.

```
GFL_EXIF_DATA * gflBitmapGetEXIF(  
    GFL_BITMAP * bitmap,  
    GFL_UINT32  flags  
);
```

## Parameters

*bitmap*  
Pointer to a [GFL\\_BITMAP](#) structure.

*flags*  
Not used.

## Return value

The function returns a pointer to a [GFL\\_EXIF\\_DATA](#) structure.

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflFreeEXIF](#),  
[gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflBitmapSetIPTC](#), [gflFreeIPTC](#), [gflNewIPTC](#), [gflSetIPTCValue](#), [gflRemoveIPTCValue](#),  
[gflLoadIPTC](#), [gflSaveIPTC](#)

# gflFreeEXIF

The **gflFreeEXIF** function frees memory allocated by [gflBitmapGetEXIF](#) function.

```
void gflFreeEXIF(  
    GFL_EXIF_DATA * exif_data  
);
```

## Parameters

*exif\_data*

Pointer to a [GFL\\_EXIF\\_DATA](#) structure.

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#),  
[gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflBitmapSetIPTC](#), [gflFreeIPTC](#), [gflNewIPTC](#), [gflSetIPTCValue](#), [gflRemoveIPTCValue](#),  
[gflLoadIPTC](#), [gflSaveIPTC](#)

# gflBitmapHasIPTC

The **gflBitmapHasIPTC** function is used to know if the picture has IPTC metadata.

```
GFL_BOOL gflBitmapHasIPTC(  
    GFL_BITMAP * bitmap  
);
```

## Parameters

*bitmap*

Pointer to a [GFL\\_BITMAP](#) structure.

*comment*

Pointer to a null-terminated string that contains the comment.

## Return value

The function returns `GFL_TRUE` if the bitmap has IPTC metadata.

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#), [gflBitmapGetIPTC](#), [gflBitmapSetIPTC](#), [gflFreeIPTC](#), [gflNewIPTC](#), [gflSetIPTCValue](#), [gflRemoveIPTCValue](#), [gflLoadIPTC](#), [gflSaveIPTC](#)

# gflBitmapGetIPTC

The **gflBitmapGetIPTC** function returns IPTC metadata in a readable form.

```
GFL_IPTC_DATA * gflBitmapGetIPTC(  
    GFL_BITMAP * bitmap  
);
```

## Parameters

*bitmap*

Pointer to a [GFL\\_BITMAP](#) structure.

## Return value

The function returns a pointer to a [GFL\\_IPTC\\_DATA](#) structure.

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#), [gflBitmapHasIPTC](#), [gflBitmapSetIPTC](#), [gflFreeIPTC](#), [gflNewIPTC](#), [gflSetIPTCValue](#), [gflRemoveIPTCValue](#), [gflLoadIPTC](#), [gflSaveIPTC](#)

# gflNewIPTC

The **gflNewIPTC** function returns IPTC metadata in a readable form.

```
GFL_IPTC_DATA * gflNewIPTC(  
    void  
);
```

## Return value

The function returns a pointer to a [GFL\\_IPTC\\_DATA](#) structure.

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#), [gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflBitmapSetIPTC](#), [gflFreeIPTC](#), [gflSetIPTCValue](#), [gflRemoveIPTCValue](#), [gflLoadIPTC](#), [gflSaveIPTC](#)

# gflFreeIPTC

The **gflFreeIPTC** function frees memory allocated by [gflBitmapGetIPTC](#) function.

```
void gflFreeIPTC(  
    GFL_IPTC_DATA * iptc_data  
);
```

## Parameters

*iptc\_data*  
Pointer to a [GFL\\_IPTC\\_DATA](#) structure.

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#),  
[gflFreeEXIF](#), [gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#)

# gflSetIPTCValue

The **gflSetIPTCValue** function set an IPTC value.

```
GFL_ERROR gflSetIPTCValue(  
    GFL_IPTC_DATA * iptc_data,  
    GFL_UINT32 id,  
    const char * value  
);
```

## Parameters

*iptc\_data*

Pointer to a [GFL IPTC DATA](#) structure.

*id*

IPTC id to change.

*value*

Pointer to a null-terminated string that contains the new value.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#), [gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflBitmapSetIPTC](#), [gflFreeIPTC](#), [gflNewIPTC](#), [gflRemoveIPTCValue](#), [gflLoadIPTC](#), [gflSaveIPTC](#)

# gflRemoveIPTCValue

The **gflRemoveIPTCValue** function remove an IPTC value.

```
GFL_IPTC_DATA * gflRemoveIPTCValue(  
    GFL_IPTC_DATA * iptc_data,  
    GFL_UINT32 id  
);
```

## Parameters

`iptc_data`  
Pointer to a [GFL\\_IPTC\\_DATA](#) structure.

`id`  
IPTC id to remove.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#),  
[gflFreeEXIF](#), [gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflBitmapSetIPTC](#), [gflFreeIPTC](#), [gflNewIPTC](#), [gflSetIPTCValue](#),  
[gflLoadIPTC](#), [gflSaveIPTC](#)

# gflLoadIPTC

The **gflLoadIPTC** allows to load IPTC data from a picture file, without loading it.

```
GFL_IPTC_DATA * gflLoadIPTC(  
    const char * filename,  
);
```

## Parameters

filename

Pointer to a null-terminated string that contains the filename to extract IPTC. Must be a JPEG file.

## Return value

The function returns a pointer to a [GFL\\_IPTC\\_DATA](#) structure.

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#), [gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflBitmapSetIPTC](#), [gflFreeIPTC](#), [gflNewIPTC](#), [gflSetIPTCValue](#), [gflRemoveIPTCValue](#), [gflSaveIPTC](#)

# gflSaveIPTC

The **gflSaveIPTC** allows to save IPTC data into a picture file, without loading it.

```
GFL_ERROR gflSaveIPTC(  
    const char * filename,  
    const GFL_IPTC_DATA * iptc_data,  
);
```

## Parameters

filename

Pointer to a null-terminated string that contains the filename to save. Must be a JPEG file.

iptc\_data

Pointer to a [GFL IPTC DATA](#) structure.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#), [gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflBitmapSetIPTC](#), [gflFreeIPTC](#), [gflNewIPTC](#), [gflSetIPTCValue](#), [gflRemoveIPTCValue](#), [gflLoadIPTC](#)

# gflBitmapSetIPTC

The **gflBitmapSetIPTC** function sets IPTC metadata to a bitmap.

```
GFL_ERROR gflBitmapSetIPTC(  
    GFL_BITMAP * bitmap,  
    const GFL_IPTC_DATA * iptc_data  
);
```

## Parameters

*bitmap*

Pointer to a [GFL\\_BITMAP](#) structure.

*iptc\_data*

Pointer to a [GFL\\_IPTC\\_DATA](#) structure.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflBitmapSetComment](#), [gflBitmapRemoveMetaData](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#), [gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflFreeIPTC](#), [gflNewIPTC](#), [gflSetIPTCValue](#), [gflRemoveIPTCValue](#), [gflLoadIPTC](#), [gflSaveIPTC](#)

# gflBitmapRemoveMetadata

The **gflBitmapRemoveMetadata** function remove all metadata of a picture.

```
void gflBitmapRemoveMetadata(  
    GFL_BITMAP * bitmap  
);
```

## Parameters

*bitmap*

Pointer to a [GFL\\_BITMAP](#) structure.

## See also

[gflBitmapSetComment](#), [gflBitmapHasEXIF](#), [gflBitmapRemoveEXIFThumbnail](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#),  
[gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflFreeIPTC](#)

# gflResize

The **gflResize** function allows to resize a picture.

```
GFL_ERROR gflResize(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    GFL_INT32 width,  
  
    GFL_INT32 height,  
  
    GFL_UINT32 method,  
  
    GFL_UINT32 flags  
);
```

## Parameters

**src**  
Pointer to a [GFL\\_BITMAP](#) structure.

**dst**  
Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

**width**  
New width.

**height**  
New height.

**method**  
GFL\_RESIZE\_QUICK Quick  
GFL\_RESIZE\_BILINEAR Bilinear

**flags**  
Reserved, must be 0.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflResizeCanvas](#), [gflCrop](#), [gflRotate](#), [gflRotateFine](#), [gflFlipHorizontal](#), [gflFlipVertical](#)

# gflResizeCanvas

The **gflResizeCanvas** function allows to resize the canvas of a picture.

```
GFL_ERROR gflResizeCanvas(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    GFL_INT32 width,  
  
    GFL_INT32 height,  
  
    GFL_CANVASRESIZE mode,  
  
    const GFL_COLOR * color  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

width

New width.

height

New height.

mode

GFL_CANVASRESIZE_CENTER	Center
GFL_CANVASRESIZE_TOPLEFT	Top-Left
GFL_CANVASRESIZE_TOPRIGHT	Top-Right
GFL_CANVASRESIZE_BOTTOMLEFT	Bottom-Left
GFL_CANVASRESIZE_BOTTOMRIGHT	Bottom-Right

color

Pointer to a [GFL\\_COLOR](#) structure to receive the

background color.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflResize](#), [gflCrop](#), [gflRotate](#), [gflRotateFine](#), [gflFlipHorizontal](#), [gflFlipVertical](#)

# gflCrop

The **gflCrop** function crop a picture.

```
GFL_ERROR gflCrop(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    const GFL_RECT * rect  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- dst  
Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.
- rect  
Crop rectangle.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflResize](#), [gflResizeCanvas](#), [gflRotate](#), [gflRotateFine](#), [gflFlipHorizontal](#), [gflFlipVertical](#)

# gflAutoCrop

The **gflAutoCrop** function performs a automatic crop on a picture.

```
GFL_ERROR gflAutoCrop(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    const GFL_COLOR * color,  
  
    GFL_INT32 tolerance  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

color

Pointer to a [GFL\\_COLOR](#) structure used to set the background color to search.  
Can be NULL, the background color is the color at x=0, y=0.

tolerance

Color tolerance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflCrop](#), [gflResize](#), [gflResizeCanvas](#), [gflRotate](#), [gflRotateFine](#), [gflFlipHorizontal](#), [gflFlipVertical](#)

# gflRotate

The **gflRotate** function applies a rotation on a picture.

```
GFL_ERROR gflRotate(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    GFL_INT32 angle,  
    const GFL_COLOR * color  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

angle

Angle of rotation in degrees.

color

Pointer to a [GFL\\_COLOR](#) structure used to set the  
background color.  
Can be NULL, the background color is (0,0,0).

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a  
value of [GFL\\_ERROR](#).

## See also

[gflResize](#), [gflResizeCanvas](#), [gflCrop](#), [gflFlipHorizontal](#), [gflFlipVertical](#), [gflRotateFine](#),

# gflRotateFine

The **gflRotateFine** function applies a rotation on a picture.

```
GFL_ERROR gflRotateFine(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    double angle,  
    const GFL_COLOR * color  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

angle

Angle of rotation in degrees.

color

Pointer to a [GFL\\_COLOR](#) structure used to set the  
background color.  
Can be NULL, the background color is (0,0,0).

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a  
value of [GFL\\_ERROR](#).

## See also

[gflResize](#), [gflResizeCanvas](#), [gflCrop](#), [gflFlipHorizontal](#), [gflFlipVertical](#), [gflRotate](#),

# gflFlipHorizontal

The **gflFlipHorizontal** function applies a horizontal flip on picture.

```
GFL_ERROR gflFlipHorizontal(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflResize](#), [gflResizeCanvas](#), [gflCrop](#), [gflRotate](#), [gflRotateFine](#), [gflFlipVertical](#)

# gflFlipVertical

The **gflFlipVertical** function applies a vertical flip on picture.

```
GFL_ERROR gflFlipVertical(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflResize](#), [gflResizeCanvas](#), [gflCrop](#), [gflRotate](#), [gflRotateFine](#), [gflFlipHorizontal](#)

# gflChangeColorDepth

The **gflChangeColorDepth** function changes the picture type.

```
GFL_ERROR gflChangeColorDepth(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    GFL_MODE mode,  
  
    GFL_MODE_PARAMS params  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

mode

GFL_MODE_TO_BINARY	Binary (8 bits)
GFL_MODE_TO_4GREY	4 Greyscale (8 bits)
GFL_MODE_TO_8GREY	8 Greyscale (8 bits)
GFL_MODE_TO_16GREY	16 Greyscale (8 bits)
GFL_MODE_TO_32GREY	32 Greyscale (8 bits)
GFL_MODE_TO_64GREY	64 Greyscale (8 bits)
GFL_MODE_TO_128GREY	128 Greyscale (8 bits)
GFL_MODE_TO_216GREY	216 Greyscale (8 bits)
GFL_MODE_TO_256GREY	256 Greyscale (8 bits)
GFL_MODE_TO_8COLORS	8 Colors (8 bits)
GFL_MODE_TO_16COLORS	16 Colors (8 bits)
GFL_MODE_TO_32COLORS	32 Colors (8 bits)
GFL_MODE_TO_64COLORS	64 Colors (8 bits)
GFL_MODE_TO_128GREY	128 Colors (8 bits)

GFL_MODE_TO_216COLORS	216 Colors (8 bits)
GFL_MODE_TO_256COLORS	256 Colors (8 bits)
GFL_MODE_TO_RGB	Red-Green-Blue (24 bits)
GFL_MODE_TO_RGBA	Red-Green-Blue-Alpha (32 bits)
GFL_MODE_TO_BGR	Blue-Green-Red (24 bits)
GFL_MODE_TO_ABGR	Alpha-Blue-Green-Red (32 bits)
GFL_MODE_TO_BGRA	Blue-Green-Red-Alpha (32 bits)

params

Indicates a dither to be used for colors, greyscale & binary.

GFL_MODE_NO_DITHER	No dithering
GFL_MODE_ADAPTIVE	Adaptive without dithering
GFL_MODE_PATTERN_DITHER	Pattern dithering
GFL_MODE_HALFTONE45_DITHER	HalfTone 45 dithering
GFL_MODE_HALFTONE90_DITHER	HalfTone 90 dithering
GFL_MODE_FLOYD_STEINBERG	Floyd-Steinberg dithering

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetColorAt](#), [gflGetNumberOfColorsUsed](#)

# gflMerge

The **gflMerge** function allows to merge a list of picture.

```
GFL_ERROR gflMerge(  
    const GFL_BITMAP * src[],  
    const GFL_POINT origin[],  
    const GFL_UINT32 opacity[],  
    GFL_INT32 num_bitmap,  
    GFL_BITMAP ** dst  
);
```

## Parameters

- src  
Address of an array of pointer to [GFL\\_BITMAP](#) structure.
- origin  
Address of an array of [GFL\\_POINT](#) structure, origin to insert for each picture.  
If NULL, origin used is (0,0).
- opacity  
Address of an array of opacity, for each picture.
- num\_bitmap  
Number of picture to merge.
- dst  
Address of a pointer to a [GFL\\_BITMAP](#) structure.

## Remark

gflMerge works only in 24 or 32bits.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

# gflBitblt

The **gflBitblt** function performs a block transfert between two pictures.

```
GFL_ERROR gflBitblt(  
    const GFL_BITMAP * src,  
  
    const GFL_RECT * rect,  
  
    const GFL_BITMAP * dst,  
  
    GFL_INT32 x_dest,  
  
    GFL_INT32 y_dest  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure, used as source.
- rect  
Pointer to a [GFL\\_RECT](#) structure. Area to copy.
- dst  
Pointer to a [GFL\\_BITMAP](#) structure, used as destination.
- x\_dest  
X position in the destination picture.
- y\_dest  
Y position in the destination picture.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

# gflBitblt

The **gflBitblt** function performs a block transfert between two pictures with alpha blending.

```
GFL_ERROR gflBitbltEx(  
    const GFL_BITMAP * src,  
  
    const GFL_RECT * rect,  
  
    const GFL_BITMAP * dst,  
  
    GFL_INT32 x_dest,  
  
    GFL_INT32 y_dest  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure, used as source. Must be 32bits.
- rect  
Pointer to a [GFL\\_RECT](#) structure. Area to copy.
- dst  
Pointer to a [GFL\\_BITMAP](#) structure, used as destination.
- x\_dest  
X position in the destination picture.
- y\_dest  
Y position in the destination picture.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

# gflGetColorAt

The **gflGetColorAt** function gets the color at a position of the picture.

```
GFL_ERROR gflGetColorAt(  
    GFL_BITMAP * src,  
  
    GFL_INT32 x,  
  
    GFL_INT32 y,  
  
    GFL_COLOR * color  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- x  
X position.
- y  
Y position.
- color  
Pointer to a [GFL\\_COLOR](#) structure to obtain the result.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetNumberOfColorsUsed](#), [gflSetColorAt](#)

# gflSetColorAt

The **gflSetColorAt** function allows to set a color at a position of the picture.

```
GFL_ERROR gflSetColorAt(  
    GFL_BITMAP * src,  
    GFL_INT32 x,  
    GFL_INT32 y,  
    const GFL_COLOR * color  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- x  
X position.
- y  
Y position.
- color  
Pointer to a [GFL\\_COLOR](#) structure.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetNumberOfColorsUsed](#), [gflGetColorAt](#)

# gflBrightness

The **gflBrightness** function increase or decrease the brightness of a picture.

```
GFL_ERROR gflBrightness(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    GFL_INT32 brightness  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

brightness

An integer between -255 and 255.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflContrast](#), [gflGamma](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualize](#), [gflEqualizeOnLuminance](#), [gflBalance](#),  
[gflAdjust](#), [gflAdjustHLS](#)

# gflContrast

The **gflContrast** function increase or decrease the contrast of a picture.

```
GFL_ERROR gflContrast(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    GFL_INT32 contrast  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

contrast

An integer between -127 and 127.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighthness](#), [gflGamma](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualize](#), [gflEqualizeOnLuminance](#), [gflBalance](#),  
[gflAdjust](#), [gflAdjustHLS](#)

# gflGamma

The **gflGamma** function increase or decrease the gamma of a picture.

```
GFL_ERROR gflGamma(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    double gamma  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

gamma

A number between 0.01 and 5.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighthness](#), [gflContrast](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualize](#), [gflEqualizeOnLuminance](#), [gflBalance](#),  
[gflAdjust](#), [gflAdjustHLS](#)

# gflAdjust

The **gflAdjust** function allows to adjust brightness, contrast & gamma of a picture.

```
GFL_ERROR gflAdjust(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    GFL_INT32 brightness,  
  
    GFL_INT32 contrast,  
  
    double gamma  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- dst  
Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.
- brightness  
An integer between -255 and 255.
- contrast  
An integer between -127 and 127.
- gamma  
A number between 0.01 and 5.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighness](#), [gflContrast](#), [gflGamma](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualize](#), [gflEqualizeOnLuminance](#),

[gflAdjustHLS](#),

# gflAdjustHLS

The **gflAdjustHLS** function allows to adjust the hue, lightness & saturation of a picture.

```
GFL_ERROR gflAdjustHLS(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    GFL_INT32 h_increment,  
  
    GFL_INT32 l_increment,  
  
    GFL_INT32 s_increment  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- dst  
Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.
- h\_increment  
An interger between -100 and 100 to add to the hue value.
- l\_increment  
An interger between -100 and 100 to add to the lightness value.
- s\_increment  
An interger between -100 and 100 to add to the saturation value.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighthness](#), [gflContrast](#), [gflGamma](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualize](#), [gflEqualizeOnLuminance](#),  
[gflAdjust](#),

# gflNegative

The **gflNegative** function applies the negative of a picture.

```
GFL_ERROR gflNegative(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP * dst  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflBrightness](#), [gflContrast](#), [gflGamma](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualize](#), [gflEqualizeOnLuminance](#), [gflBalance](#),  
[gflAdjust](#), [gflAdjustHLS](#)

# gflLogCorrection

The **gflLogCorrection** function applies a logarithmic correction on a picture.

```
GFL_ERROR gflLogCorrection(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighthness](#), [gflContrast](#), [gflGamma](#), [gflNormalize](#), [gflEqualize](#), [gflEqualizeOnLuminance](#), [gflBalance](#), [gflAdjust](#), [gflAdjustHLS](#)

# gflNormalize

The **gflNormalize** function applies a normalisation of the pixels values.

```
GFL_ERROR gflNormalize(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighness](#), [gflContrast](#), [gflGamma](#), [gflLogCorrection](#), [gflEqualize](#), [gflEqualizeOnLuminance](#), [gflBalance](#),  
[gflAdjust](#), [gflAdjustHLS](#)

# gflEqualize

The **gflEqualize** function applies an equalization of the pixels.

```
GFL_ERROR gflEqualize(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighness](#), [gflContrast](#), [gflGamma](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualizeOnLuminance](#), [gflBalance](#),  
[gflAdjust](#), [gflAdjustHLS](#)

# gflEqualizeOnLuminance

The **gflEqualizeOnLuminance** function applies an equalization of the pixels (based on the luminance).

```
GFL_ERROR gflEqualizeOnLuminance(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighness](#), [gflContrast](#), [gflGamma](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualize](#), [gflBalance](#), [gflAdjust](#), [gflAdjustHLS](#)

# gflBalance

The **gflBalance** function applies a color balance of a picture.

```
GFL_ERROR gflBalance(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    const GFL_COLOR * color  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- dst  
Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.
- color  
Pointer to a [GFL\\_COLOR](#) structure.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighthness](#), [gflContrast](#), [gflGamma](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualize](#) [gflEqualizeOnLuminance](#),  
[gflBalance](#), [gflAdjust](#), [gflAdjustHLS](#)

# gflSwapColors

The **gflSwapColors** function allows to swap component.

```
GFL_ERROR gflSwapColors(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    GFL_SWAPCOLORS_MODE mode  
);
```

## Parameters

**src**  
Pointer to a [GFL\\_BITMAP](#) structure.

**dst**  
Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

**mode**  
GFL\_SWAPCOLORS\_RBG 0  
GFL\_SWAPCOLORS\_BGR 1  
GFL\_SWAPCOLORS\_BRG 2  
GFL\_SWAPCOLORS\_GRB 3  
GFL\_SWAPCOLORS\_GBR 4

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

# gflSepia

The **gflSepia** function converts a picture in sepia.

```
GFL_ERROR gflSepia(  
    GFL_BITMAP * src,  
    GFL_BITMAP ** dst,  
    GFL_INT32 percent  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

percent

An integer between 0 and 100.

0 => greyscale, 100 => maximum sepia

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflSepiaExt](#)

# gflSepiaEx

The **gflSepiaEx** function converts a picture in sepia.

```
GFL_ERROR gflSepiaEx(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst,  
  
    GFL_INT32 percent,  
  
    const GFL_COLOR * color  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- dst  
Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.
- percent  
An integer between 0 and 100.  
0 => greyscale, 100 => maximum sepia
- color  
Pointer to a [GFL\\_COLOR](#) structure.  
This color is used as a reference.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflSepia](#)

# gflAutomaticLevels

The **gflAutomaticLevels** function applies an automatic equalisation of levels.

```
GFL_ERROR gflAutomaticLevels(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighness](#), [gflContrast](#), [gflGamma](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualizeOnLuminance](#), [gflBalance](#),  
[gflAdjust](#), [gflAdjustHLS](#), [gflAutomaticContrast](#)

# gflAutomaticContrast

The **gflAutomaticContrast** function adjusts the contrast of picture.

```
GFL_ERROR gflAutomaticContrast(  
    GFL_BITMAP * src,  
  
    GFL_BITMAP ** dst  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflNegative](#), [gflBrighness](#), [gflContrast](#), [gflGamma](#), [gflLogCorrection](#), [gflNormalize](#), [gflEqualizeOnLuminance](#), [gflBalance](#),  
[gflAdjust](#), [gflAdjustHLS](#) [gflAutomaticLevels](#)

Applies a filter on a picture.

To do.

# gflConvolve

The **gflConvolve** function applies a convolution matrix on a picture.

```
GFL_ERROR gflConvolve(  
    GFL_BITMAP * src,  
    GFL_BITMAP ** dst,  
    const GFL_FILTER * filter  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- dst  
Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.
- filter  
Pointer to a [GFL\\_FILTER](#) structure.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

# gflGetNumberOfColorsUsed

The **gflGetNumberOfColorsUsed** gets the total unique colors of a picture.

```
GFL_UINT32 gflGetNumberOfColorsUsed(  
    GFL_BITMAP * src  
);
```

## Parameters

src  
Pointer to a [GFL\\_BITMAP](#) structure.

## Return value

Total unique colors.

## See also

[gflChangeColorDepth](#)

# gflJpegLosslessTransform

The **gflJpegLosslessTransform** function applies lossless transformations on a JPEG file.

```
GFL_ERROR gflJpegLosslessTransform(  
    const char * filename,  
  
    GFL_LOSSLESS_TRANSFORM transform  
);
```

## Parameters

filename

Pointer to a null-terminated string that contains the filename to modify.

transform

Transformation

GFL_LOSSLESS_TRANSFORM_ROTATE90	Rotate of 90 degrees
GFL_LOSSLESS_TRANSFORM_ROTATE180	Rotate of 180 degrees
GFL_LOSSLESS_TRANSFORM_ROTATE270	Rotate of 270 degrees
GFL_LOSSLESS_TRANSFORM_VERTICAL_FLIP	Vertical flip
GFL_LOSSLESS_TRANSFORM_HORIZONTAL_FLIP	Horizontal flip

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

# gflConvertBitmapIntoDIB

The **gflConvertBitmapIntoDIB** function converts a [GFL\\_BITMAP](#) in a Windows Device Independent Bitmap.

```
GFL_ERROR gflConvertBitmapIntoDIB(  
    const GFL_BITMAP * bitmap,  
  
    HANDLE * hDIB  
);
```

## Parameters

*bitmap*  
Pointer to a [GFL\\_BITMAP](#) structure.

*hDIB*  
Address of a DIB HANDLE.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflConvertBitmapIntoDDB](#), [gflConvertDIBIntoBitmap](#), [gflConvertDDBIntoBitmap](#), [gflLoadBitmapIntoDIB](#), [gflLoadBitmapIntoDDB](#), [gflAddText](#)

# gflConvertBitmapIntoDDB

The **gflConvertBitmapIntoDDB** function converts a [GFL\\_BITMAP](#) in a Windows Device Dependant Bitmap.

```
GFL_ERROR gflConvertBitmapIntoDIB(  
    const GFL_BITMAP * bitmap,  
  
    HBITMAP * hBitmap  
);
```

## Parameters

*bitmap*  
Pointer to a [GFL\\_BITMAP](#) structure.

*hBitmap*  
Address of a HBITMAP.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflConvertBitmapIntoDIB](#), [gflConvertDIBIntoBitmap](#), [gflConvertDDBIntoBitmap](#), [gflLoadBitmapIntoDIB](#), [gflLoadBitmapIntoDDB](#), [gflAddText](#)

# gflConvertDIBIntoBitmap

The **gflConvertDIBIntoBitmap** function converts a Windows Device Independent Bitmap into [GFL\\_BITMAP](#).

```
GFL_ERROR gflConvertDIBIntoBitmap(  
    HANDLE hDIB,  
  
    GFL_BITMAP ** bitmap  
);
```

## Parameters

hDIB

A HANDLE on the DIB.

bitmap

Address of a pointer to a [GFL\\_BITMAP](#) structure.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflConvertBitmapIntoDIB](#), [gflConvertBitmapIntoDDB](#), [gflConvertDDBIntoBitmap](#), [gflLoadBitmapIntoDIB](#), [gflLoadBitmapIntoDDB](#), [gflAddText](#)

# gflConvertDDBIntoBitmap

The **gflConvertDDBIntoBitmap** function converts a Windows Device Dependant Bitmap into [GFL\\_BITMAP](#).

```
GFL_ERROR gflConvertDDBIntoBitmap(  
    HBITMAP hBitmap,  
  
    GFL_BITMAP ** bitmap  
);
```

## Parameters

hBitmap

A HANDLE on the HBITMAP.

bitmap

Address of a pointer to a [GFL\\_BITMAP](#) structure.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflConvertBitmapIntoDIB](#), [gflConvertBitmapIntoDDB](#), [gflConvertDIBIntoBitmap](#), [gflLoadBitmapIntoDIB](#), [gflLoadBitmapIntoDDB](#), [gflAddText](#)

# gflLoadBitmapIntoDIB

The **gflLoadBitmapIntoDIB** function load a picture file into a Windows Device Independant Bitmap.

```
GFL_ERROR gflLoadBitmapIntoDIB(  
    const char * filename,  
    HANDLE * hDIB,  
    GFL_LOAD_PARAMS * params,  
    GFL_FILE_INFORMATION * informations  
);
```

## Parameters

filename

Pointer to a null-terminated string that contains the filename to load.

hDIB

Address of a DIB HANDLE.

params

Pointer to a [GFL\\_LOAD\\_PARAMS](#) structure.  
This structure must be filled correctly.

informations

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure. Can be NULL if you don't want it.  
You must use [gflFreeInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflConvertBitmapIntoDIB](#), [gflConvertBitmapIntoDDB](#), [gflConvertDIBIntoBitmap](#), [gflConvertDDBIntoBitmap](#), [gflLoadBitmapIntoDDB](#), [gflAddText](#)

# gflLoadBitmapIntoDDB

The **gflLoadBitmapIntoDDB** function load a picture file into a Windows Device Dependant Bitmap.

```
GFL_ERROR gflLoadBitmapIntoDDB(  
    const char * filename,  
    HBITMAP * hBitmap,  
    GFL_LOAD_PARAMS * params,  
    GFL_FILE_INFORMATION * informations  
);
```

## Parameters

filename

Pointer to a null-terminated string that contains the filename to load.

hBitmap

Address of a HBITMAP.

params

Pointer to a [GFL\\_LOAD\\_PARAMS](#) structure. This structure must be filled correctly.

informations

Pointer to a [GFL\\_FILE\\_INFORMATION](#) structure. Can be NULL if you don't want it. You must use [gflFreeInformation](#) to free his content.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflConvertBitmapIntoDIB](#), [gflConvertBitmapIntoDDB](#), [gflConvertDIBIntoBitmap](#), [gflConvertDDBIntoBitmap](#), [gflLoadBitmapIntoDIB](#), [gflAddText](#)

# gflAddText

The **gflAddText** function adds a text on a [GFL\\_BITMAP](#).

```
GFL_ERROR gflAddText(  
    GFL_BITMAP * bitmap,  
    const char * text,  
    const char * font_name,  
    GFL_INT32 x,  
    GFL_INT32 y,  
    GFL_INT32 font_size,  
    GFL_INT32 orientation,  
    GFL_BOOL italic,  
    GFL_BOOL bold,  
    GFL_BOOL strike_out,  
    GFL_BOOL underline,  
    GFL_BOOL antialias,  
    const GFL_COLOR * color  
);
```

## Parameters

*bitmap*

Pointer to a [GFL\\_BITMAP](#) structure.

*text*

Pointer to a null-terminated string that contains the text to add.

*font\_name*

Pointer to a null-terminated string that contains the name of the font to use.

*x*

X position.

*y*

Y position.

*font\_size*

Height of the font.

*orientation*

Orientation of the text (degrees).

*italic*

Specifies a italic font.  
**bold**  
Specifies a bold font.  
**strike\_out**  
Specifies a strikethrough font.  
**underline**  
Specifies a underline font.  
**antialias**  
Font is antialiased.  
**color**  
Pointer to a [GFL\\_COLOR](#) structure for the text color.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflConvertBitmapIntoDIB](#), [gflConvertBitmapIntoDDB](#), [gflConvertDIBIntoBitmap](#), [gflConvertDDBIntoBitmap](#), [gflLoadBitmapIntoDIB](#), [gflLoadBitmapIntoDDB](#)

# gflImportFromClipboard

The **gflImportFromClipboard** function allows to import the picture from the clipboard. .

```
GFL_ERROR gflImportFromClipboard(  
    GFL_BITMAP ** bitmap  
);
```

## Parameters

*bitmap*

Address of a pointer to a [GFL\\_BITMAP](#) structure.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflExportFromClipboard](#), [gflImportFromHWND](#)

# gflExportIntoClipboard

The **gflExportIntoClipboard** function allows to export a picture into clipboard..

```
GFL_ERROR gflExportIntoClipboard(  
    GFL_BITMAP * bitmap  
);
```

## Parameters

*bitmap*

Pointer to a [GFL\\_BITMAP](#) structure.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflImportFromClipboard](#), [gflImportFromHWND](#)

# gflImportFromHWND

The **gflImportFromHWND** function allows to copy the content of a window. .

```
GFL_ERROR gflImportFromHWND(  
    HWND hwnd  
  
    const GFL_RECT * rect,  
  
    GFL_BITMAP ** bitmap  
);
```

## Parameters

- `hwnd`  
Handle of the window.
- `rect`  
Rectangle to copy.  
Can be NULL.
- `bitmap`  
Address of a pointer to a [GFL\\_BITMAP](#) structure.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflImportFromClipboard](#), [gflExportFromClipboard](#)

# gflDrawPointColor

The **gflDrawPointColor** function allows to draw a point on a picture.

```
GFL_ERROR gflDrawPointColor(  
    GFL_BITMAP * src,  
  
    GFL_INT32 x,  
  
    GFL_INT32 y,  
  
    GFL_UINT32 line_width,  
  
    const GFL_COLOR * line_color,  
  
    GFL_BITMAP ** dst,  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- x  
X position.
- y  
Y position.
- line\_width  
Width of the point (1 à 13).
- line\_color  
Pointer to a [GFL\\_COLOR](#) structure. Color of the point.
- dst  
Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflDrawLineColor](#), [gflDrawPolylineColor](#), [gflDrawRectangleColor](#), [gflDrawPolygonColor](#), [gflDrawCircleColor](#)

# gflDrawLineColor

The **gflDrawLineColor** function allows to draw a line on a picture.

```
GFL_ERROR gflDrawLineColor(  
    GFL_BITMAP * src,  
  
    GFL_INT32 x0,  
  
    GFL_INT32 y0,  
  
    GFL_INT32 x1,  
  
    GFL_INT32 y1,  
  
    GFL_UINT32 line_width,  
  
    const GFL_COLOR * line_color,  
  
    GFL_LINE_STYLE line_style,  
  
    GFL_BITMAP ** dst,  
);
```

## Parameters

src  
 Pointer to a [GFL\\_BITMAP](#) structure.

x0  
 X start position.

y0  
 Y start position.

x1  
 X end position.

y1  
 Y end position.

line\_width  
 Width of the line (1 to 13).

line\_color

Pointer to a [GFL\\_COLOR](#) structure. Color of the line.

line\_style

Works only with a line width of 1.

GFL_LINE_STYLE_SOLID	Solid
GFL_LINE_STYLE_DASH	Dashes
GFL_LINE_STYLE_DOT	Dots
GFL_LINE_STYLE_DASHDOT	Alternating dashes and dots
GFL_LINE_STYLE_DASHDOTDOT	Alternating dashes and double dots

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflDrawPointColor](#), [gflDrawPolylineColor](#), [gflDrawRectangleColor](#), [gflDrawPolygonColor](#), [gflDrawCircleColor](#)

# gflDrawPolylineColor

The **gflDrawPolylineColor** function allows to draw a polyline on a picture.

```
GFL_ERROR gflDrawPolylineColor(  
    GFL_BITMAP * src,  
  
    const GFL_POINT points[],  
  
    GFL_INT32 num_points,  
  
    GFL_UINT32 line_width,  
  
    const GFL_COLOR * line_color,  
  
    GFL_LINE_STYLE line_style,  
  
    GFL_BITMAP ** dst,  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

points

Address of a array of [GFL\\_POINT](#) structure.

num\_points

Number of points.

line\_width

Width fo the line (1 to 13).

line\_color

Pointer to a [GFL\\_COLOR](#) structure.

line\_style

Works only with a line width of 1.

GFL\_LINE\_STYLE\_SOLID           Solid

GFL\_LINE\_STYLE\_DASH           Dashes

GFL\_LINE\_STYLE\_DOT           Dots

GFL\_LINE\_STYLE\_DASHDOT       Alternating dashes and

`GFL_LINE_STYLE_DASHDOTDOT` dots  
Alternating dashes and double dots

`dst`

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflDrawPointColor](#), [gflDrawLineColor](#), [gflDrawRectangleColor](#), [gflDrawPolygonColor](#), [gflDrawCircleColor](#)

# gflDrawRectangleColor

The **gflDrawRectangleColor** function allows to draw a rectangle on a picture.

```
GFL_ERROR gflDrawRectangleColor(  
    GFL_BITMAP * src,  
  
    GFL_INT32 x,  
  
    GFL_INT32 y,  
  
    GFL_INT32 width,  
  
    GFL_INT32 height,  
  
    const GFL_COLOR * fill_color,  
  
    GFL_UINT32 line_width,  
  
    const GFL_COLOR * line_color,  
  
    GFL_LINE_STYLE line_style,  
  
    GFL_BITMAP ** dst,  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- x  
X start.
- y  
Y start.
- width  
Width of the rectangle.
- height  
Height of the rectangle.
- fill\_color  
Pointer of a [GFL\\_COLOR](#) structure.

If NULL, no fill.

**line\_width**  
Width of the line (1 to 13).

**line\_color**  
Pointer of a [GFL\\_COLOR](#) structure.  
If NULL, no outline.

**line\_style**  
Works only with a line width of 1.

<code>GFL_LINE_STYLE_SOLID</code>	Solid
<code>GFL_LINE_STYLE_DASH</code>	Dashes
<code>GFL_LINE_STYLE_DOT</code>	Dots
<code>GFL_LINE_STYLE_DASHDOT</code>	Alternating dashes and dots
<code>GFL_LINE_STYLE_DASHDOTDOT</code>	Alternating dashes and double dots

**dst**  
Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns `GFL_NO_ERROR` if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflDrawPointColor](#), [gflDrawLineColor](#), [gflDrawPolylineColor](#), [gflDrawPolygonColor](#), [gflDrawCircleColor](#)

# gflDrawPolygonColor

The **gflDrawPolygonColor** function allows to draw a poylgone on the picture.

```
GFL_ERROR gflDrawPolygonColor(  
    GFL_BITMAP * src,  
  
    const GFL_POINT points[],  
  
    GFL_INT32 num_points,  
  
    const GFL_COLOR * fill_color,  
  
    GFL_UINT32 line_width,  
  
    const GFL_COLOR * line_color,  
  
    GFL_LINE_STYLE line_style,  
  
    GFL_BITMAP ** dst,  
);
```

## Parameters

src

Pointer to a [GFL\\_BITMAP](#) structure.

points

Address to an array of [GFL\\_POINT](#) structure.

Closing the polygone is not necessary.

num\_points

Number of points.

fill\_color

Pointer to a [GFL\\_COLOR](#) structure.

If NULL, no fill.

line\_width

Width of the line (1 to 13).

line\_color

Pointer to a [GFL\\_COLOR](#) structure.

If NULL, no outline.

line\_style

Works only with a line width of 1.

GFL_LINE_STYLE_SOLID	Solid
GFL_LINE_STYLE_DASH	Dashes
GFL_LINE_STYLE_DOT	Dots
GFL_LINE_STYLE_DASHDOT	Alternating dashes and dots
GFL_LINE_STYLE_DASHDOTDOT	Alternating dashes and double dots

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.  
NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflDrawPointColor](#), [gflDrawLineColor](#), [gflDrawPolylineColor](#), [gflDrawRectangleColor](#), [gflDrawCircleColor](#)

# gflDrawCircleColor

The **gflDrawCircleColor** function allows to draw a circle on a picture.

```
GFL_ERROR gflDrawCircleColor(  
    GFL_BITMAP * src,  
  
    GFL_INT32 x,  
  
    GFL_INT32 y,  
  
    GFL_INT32 radius,  
  
    const GFL_COLOR * fill_color,  
  
    GFL_UINT32 line_width,  
  
    const GFL_COLOR * line_color,  
  
    GFL_LINE_STYLE line_style,  
  
    GFL_BITMAP ** dst,  
);
```

## Parameters

- src  
Pointer to a [GFL\\_BITMAP](#) structure.
- x  
X center.
- y  
Y center.
- radius  
Radius of the circle.
- fill\_color  
Pointer to a [GFL\\_COLOR](#) structure.  
If NULL, no fill.
- line\_width  
Width of the line (1 to 13).

line\_color

Pointer to a [GFL\\_COLOR](#) structure.

If NULL, no outline.

line\_style

Works only with a line width of 1.

GFL\_LINE\_STYLE\_SOLID           Solid

GFL\_LINE\_STYLE\_DASH           Dashes

GFL\_LINE\_STYLE\_DOT           Dots

GFL\_LINE\_STYLE\_DASHDOT       Alternating dashes and  
dots

GFL\_LINE\_STYLE\_DASHDOTDOT   Alternating dashes and  
double dots

dst

Address of a pointer to a [GFL\\_BITMAP](#) structure.

NULL if on the same instance.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflDrawPointColor](#), [gflDrawLineColor](#), [gflDrawPolylineColor](#), [gflDrawPolygonColor](#), [gflDrawRectangleColor](#),

# GFL\_BITMAP

The **GFL\_BITMAP** structure contains all informations about picture in memory.

```
typedef struct (  
    GFL_BITMAP_TYPE Type,  
    GFL_ORIGIN Origin,  
    GFL_INT32 Width,  
    GFL_INT32 Height,  
    GFL_UINT32 BytesPerLine,  
    GFL_INT16 LinePadding,  
    GFL_UINT16 BitsPerComponent,  
    GFL_UINT16 ComponentsPerPixel,  
    GFL_UINT16 BytesPerPixel,  
    GFL_UINT16 Xdpi,  
    GFL_UINT16 Ydpi,  
    GFL_INT16 TransparentIndex,  
    GFL_INT32 ColorUsed,  
    GFL_COLORMAP * ColorMap,  
    GFL_UINT8 * Data,  
    char * Comment,  
    void * MetaData  
} GFL_BITMAP
```

## Members

Type

Type of the picture

GFL_BINARY	0x0001	Binary
GFL_GREY	0x0002	Grey scale
GFL_COLORS	0x0004	Colors with colormap
GFL_RGB	0x00010	TrueColors - Red/Green/Blue
GFL_RGBA	0x0020	TrueColors - Red/Green/Blue/Alpha
GFL_BGR	0x0040	TrueColors - Blue/Green/Red
GFL_ABGR	0x0080	TrueColors - Alpha/Blue/Green/Red
		TrueColors -

GFL_BGRA	0x0100	Blue/Green/Red/Alpha
GFL_ARGB	0x0200	TrueColors - Alpha/Red/Green/Blue
GFL_CMYK	0x0400	TrueColors - Cyan/Magenta/Yellow/Black

#### Origin

Origin of the picture.

GFL_TOP_LEFT	0	Top left (default)
GFL_BOTTOM_LEFT	2	Bottom left
GFL_TOP_RIGHT	1	Top right
GFL_BOTTOM_RIGHT	3	Bottom right

#### Width

Width in pixels of the picture.

#### Height

Height in pixels of the picture.

#### BytesPerLine

Bytes per line of pixels.

#### LinePadding

Internal use, do not modify.

#### BitsPerComponent

Bits per component, can be 1, 8, 16

#### ComponentsPerPixel

Component per pixel, can be 1, 3 or 4

#### BytesPerPixel

Bytes per pixel (For example: 1, 3 or 4).

#### Xdpi

Pixels per inch in X axis.

#### Ydpi

Pixels per inch in Y axis.

#### TransparentIndex

Index of transparency (only for GFL\_COLORS & GFL\_GREY type).

#### ColorUsed

Number of color used in the picture (only for GFL\_COLORS & GFL\_GREY type).

## ColorMap

Address of a [GFL\\_COLORMAP](#) structure for the colormap (only for GFL\_COLORS type).

## Data

Pointer of the picture data.

## Comment

Address of a string used by the comment. You must use [gflSetComment](#) to change the comment.

## MetaData

Pointer of Metadata. You must use [gflBitmapGetIPTC](#) & [gflBitmapGetEXIF](#) to obtain readable data.

## See also

[gflLoadBitmap](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromHandle](#), [gflSaveBitmap](#),  
[gflSaveBitmapIntoHandle](#), [gflSetComment](#), [gflBitmapGetIPTC](#), [gflBitmapGetEXIF](#), [gflBitmapHasEXIF](#), [gflBitmapHasIPTC](#),  
[gflBitmapRemoveEXIFThumbnail](#), [gflBitmapRemoveMetaData](#)

# GFL\_COLORMAP

The **GFL\_COLORMAP** structure is used for colormap.

```
typedef struct (  
    GFL_UINT8 Red[256],  
    GFL_UINT8 Green[256],  
    GFL_UINT8 Blue[256]  
} GFL_COLORMAP
```

## Members

Red

Array of red components.

Green

Array of green components.

Blue

Array of blue components.

## See also

[GFL\\_BITMAP](#), [gflLoadBitmap](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromHandle](#), [gflSaveBitmap](#), [gflSaveBitmapIntoHandle](#)

# GFL\_FORMAT\_INFORMATION

The **GFL\_FORMAT\_INFORMATION** structure contains informations about a format available in GFL.

```
typedef struct (  
    GFL_INT32 Index,  
    GFL_ORIGIN Name[8],  
    char Description[64],  
    GFL_UINT32 Status,  
    GFL_UINT32 NumberOfExtension,  
    char Extension[16][8]  
) GFL_FORMAT_INFORMATION
```

## Members

Index

Index of the format.

Name

Null-terminated string that contains the name of the format. For example, "jpeg" is for JPEG format.

Description

Null-terminated string that contains the label of the format.

Status

Format status. .

### **Statut**

GFL\_READ      Reading support

GFL\_WRITE     Writing support

NumberOfExtension

Nombre of extension known by this format.

Extension

Array of Null-terminated string that contains the extension.

## See also

[gflGetNumberOfFormat](#), [gflGetFormatIndexByName](#), [gflGetFormatNameByIndex](#), [gflFormatsSupported](#),

[gflFormatsWritableByIndex](#), [gflFormatsWritableByName](#), [gflFormatsReadableByIndex](#), [gflFormatsReadableByName](#),  
[gflGetDefaultFormatSuffixByIndex](#), [gflGetDefaultFormatSuffixByName](#), [gflGetFormatDescriptionByIndex](#),  
[gflGetFormatDescriptionByName](#), [gflGetFormatInformationByIndex](#), [gflGetFormatInformationByName](#), [GFL\\_LOAD\\_PARAMS](#),  
[GFL\\_SAVE\\_PARAMS](#)

# GFL\_FILE\_INFORMATION

The **GFL\_FILE\_INFORMATION** structure contains informations about a picture's file.

```
typedef struct (  
    GFL_BITMAP_TYPE Type,  
    GFL_ORIGIN Origin,  
    GFL_INT32 Width,  
    GFL_INT32 Height,  
    GFL_INT32 FormatIndex,  
    char FormatName[8],  
    char Description[64],  
    GFL_UINT16 Xdpi,  
    GFL_UINT16 Ydpi,  
    GFL_UINT16 BitsPerComponent,  
    GFL_UINT16 ComponentsPerPixel,  
    GFL_INT32 NumberOfImages,  
    GFL_UINT32 FileSize,  
    GFL_COLORMODEL ColorModel,  
    GFL_COMPRESSION Compression,  
    char CompressionDescription[64]  
} GFL_LOAD_PARAMS
```

## Members

Type

Not used

Origin

Origin of the picture.

GFL_TOP_LEFT	0	Top left (default)
GFL_BOTTOM_LEFT	2	Bottom left
GFL_TOP_RIGHT	1	Top right
GFL_BOTTOM_RIGHT	3	Bottom right

Width

Width in pixels of the picture.

Height

Height in pixels of the picture.

FormatIndex

Index of picture's format.

FormatName  
Name of picture's format.

Description  
File label.

Xdpi  
Pixels per inch in the X axis.

Ydpi  
Pixels per inch in the Y axis.

BitsPerComponent  
Bits per component, can be 1, 8, 16

ComponentsPerPixel  
Component per pixel, can be 1, 3 or 4

NumberOfImages  
Number of picture in the file.

FileSize  
Size of the file.

ColorModel  
Color model.

GFL_CM_RGB	0	Red-Green-Blue
GFL_CM_GREY	1	Greyscale
GFL_CM_CMY	2	Cyan-Magenta-Yellow
GFL_CM_CMYK	3	Cyan-Magenta-Yellow-Black
GFL_CM_YCBCR	4	YCbCr
GFL_CM_YUV16	5	YUV 16bits
GFL_CM_LAB	6	Lab
GFL_CM_LOGLUV	7	Log Luv
GFL_CM_LOGL	8	Log L

Compression

GFL_NO_COMPRESSION	0	No compression
GFL_RLE	1	Packbits
GFL_LZW	2	LZW
GFL_JPEG	3	JPEG
GFL_ZIP	4	ZIP

GFL_SGI_RLE	5	GSI Packbits
GFL_CCITT_RLE	6	CCITT RLE
GFL_CCITT_FAX3	7	Fax Group 3
GFL_CCITT_FAX3_2D	8	Fax Group 3-2D
GFL_CCITT_FAX4	9	Fax Group 4
GFL_WAVELET	10	Wavelette
GFL_UNKNOWN_COMPRESSION	255	Other compression

### CompressionDescription

Pointer to a buffer that contains the full compression description.

## Remarks

gflFreeFileInformation must be used for freeing the allocated memory.

## See also

[gflFreeFileInformation](#), [gflGetFileInformation](#), [gflLoadBitmap](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromHandle](#)

# GFL\_LOAD\_PARAMS

The **GFL\_LOAD\_PARAMS** structure contains options for picture loading.

```
typedef struct (  
    GFL_UINT32 Flags,  
    GFL_INT32 FormatIndex,  
    GFL_INT32 ImageWanted,  
    GFL_ORIGIN Origin,  
    GFL_BITMAP_TYPE ColorModel,  
    GFL_UINT32 LinePadding,  
    GFL_UINT8 DefaultAlpha,  
    GFL_UINT8 Reserved1,  
    GFL_UINT16 Reserved2,  
    GFL_INT32 Width,  
    GFL_INT32 Height,  
    GFL_UINT32 Offset,  
    GFL_CORDER ChannelOrder,  
    GFL_CTYPE ChannelType,  
    GFL_UINT16 PcdBase,  
    GFL_UINT16 EpsDpi,  
    GFL_INT32 EpsWidth,  
    GFL_INT32 EpsHeight,  
    GFL_READ_CALLBACK Read,  
    GFL_TELL_CALLBACK Tell,  
    GFL_SEEK_CALLBACK Seek  
} GFL_LOAD_PARAMS
```

## Members

Flags

Options

GFL\_LOAD\_SKIP\_ALPHA

If the picture has an alpha channel it is ignored

GFL\_LOAD\_IGNORE\_READ\_ERROR

Ignore all read errors

GFL\_LOAD\_BY\_EXTENSION\_ONLY

Use only extension to recognize the

GFL_LOAD_READ_ALL_COMMENT	Read all comment in the file
GFL_LOAD_FORCE_COLOR_MODEL	ColorModel is used for the picture type
GFL_LOAD_PREVIEW_NO_CANVAS_RESIZE	Keep the ratio of the preview
GFL_LOAD_BINARY_AS_GREY	Load a binary file in 8bits
GFL_LOAD_ORIGINAL_COLORMODEL	If the color model of the file is CMYK, so the picture loaded will be in CMYK
GFL_LOAD_ONLY_FIRST_FRAME	If the color model of the file is CMYK, so the picture loaded will be in CMYK
GFL_LOAD_ORIGINAL_DEPTH	If the file has more than 8 bits per component, keep it
GFL_LOAD_METADATA	Read all metadata (IPTC, EXIF)
GFL_LOAD_COMMENT	Read comment
GFL_LOAD_HIGH_QUALITY_THUMBNAI	Use high quality for gflLoadThumbnail

### FormatIndex

Index of the format used to load.

Default value : -1 (for an automatic recognition).

## ImageWanted

For a multi-page file, identifies the image number.

Default value : 0

## Origin

Origin wanted.

GFL\_TOP\_LEFT            Top left

GFL\_BOTTOM\_LEFT        Bottom left

GFL\_TOP\_RIGHT           Top right

GFL\_BOTTOM\_RIGHT       Bottom right

Default value : GFL\_TOP\_LEFT

## ColorModel

Color Model wanted.

GFL\_RGB                 True colors - Red/Green/Blue (24 bits)

GFL\_BGR                 True colors - Blue/Green/Red (24 bits)

GFL\_RGBA                True colors - Red/Green/Blue/Alpha (32 bits)

GFL\_ABGR                True colors - Alpha/Blue/Green/Red (32 bits)

GFL\_BGRA                True colors - Blue/Green/Red/Alpha (32 bits)

GFL\_ARGB                True colors - Red/Green/Blue/Alpha (32 bits)

Default value : GFL\_RGB

## LinePadding

Pad for a pixels line (For example, a value of 4 allow a line padding on 32bits).

Default value : 1

## DefaultAlpha

Alpha value to use when the picture is loaded in 32bits, but the original file doesn't have an alpha.

Default value: Black

## Width

For RAW or YUV format, width of picture.

Height

For RAW or YUV format, height of picture.

Offset

For RAW or YUV format, offset of the picture in the file.

ChannelOrder

For RAW format, channel order of the components.

GFL\_CORDER\_INTERLEAVED Interleaved

GFL\_CORDER\_SEQUENTIAL Sequential

GFL\_CORDER\_SEPARATE Separate

ChannelType

For RAW format, channel type of the components.

GFL\_CTYPE\_GREYSCALE Greyscale

GFL\_CTYPE\_RGB Red-Green-Blue

GFL\_CTYPE\_BGR Blue-Green-Red

GFL\_CTYPE\_RGBA Red-Green-Blue-Alpha

GFL\_CTYPE\_ABGR Alpha-Blue-Green-Red

GFL\_CTYPE\_CMY Cyan-Magenta-Yellow

GFL\_CTYPE\_CMYK Cyan-Magenta-Yellow-Black

PcdBase

For PCD format, it's the base used.

0 192x144

1 384x288

2 768x576

EpsDpi

For PS/EPS format, dpi to be used for loading.

EpsWidth

For PS/EPS format, width to be used for loading.

EpsHeight

For PS/EPS format, height to be used for loading.

Read

Pointer to a read user function.

Tell

Pointer to a tell user function.

Seek

Pointer to a seek user function.

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflLoadBitmap](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromHandle](#)

# GFL\_SAVE\_PARAMS

The **GFL\_SAVE\_PARAMS** structure contains options for the save of picture.

```
typedef struct (  
    GFL_UINT32 Flags,  
    GFL_INT32 FormatIndex,  
    GFL_COMPRESSION Compression,  
    GFL_INT16 Quality,  
    GFL_INT16 CompressionLevel,  
    GFL_BOOL Interlaced,  
    GFL_BOOL Progressive,  
    GFL_BOOL OptimizeHuffmanTable,  
    GFL_BOOL InAscii,  
    GFL_UINT32 Offset,  
    GFL_CORDER ChannelOrder,  
    GFL_CTYPE ChannelType,  
    GFL_WRITE_CALLBACK Write,  
    GFL_TELL_CALLBACK Tell,  
    GFL_SEEK_CALLBACK Seek  
} GFL_SAVE_PARAMS
```

## Members

Flags

Options

<code>GFL_SAVE_REPLACE_EXTENSION</code>	Replace extension by the default format extension
<code>GFL_SAVE_ANYWAY</code>	Convert picture if colormode can be saved in this format (For example, RGB picture must be converted in 256 colors to save it in GIF)

FormatIndex

Index of format to be used.

## Compression

GFL_NO_COMPRESSION	No compression
GFL_RLE	Packbits
GFL_LZW	LZW (tiff only)
GFL_CCITT_FAX3	Fax Group 3 (tiff only)
GFL_CCITT_FAX3_2D	Fax Group 3-2D (tiff only)
GFL_CCITT_FAX4	Fax Group 4 (tiff only)

## Quality

Quality of the compression (JPEG)  
0: the worst, 100: the best

## CompressionLevel

Level of compression (PNG).  
1: minimum, 7: maximum

## Interlaced

Interlaced mode (GIF).

## Progressive

Progressive mode (JPEG).

## OptimizeHuffmanTable

Optimize the Huffman table (JPEG).

## InAscii

Use the ascii mode (PNM)

## Offset

For RAW or YUV format, offset of the data start.

## ChannelOrder

For RAW format, channel order of components.

GFL_CORDER_INTERLEAVED	Interleaved
GFL_CORDER_SEQUENTIAL	Sequential
GFL_CORDER_SEPARATE	Separate

## ChannelType

For RAW format, channel type of components.

GFL_CTYPE_GREYSCALE	Greyscale
GFL_CTYPE_RGB	Red-Green-Blue
GFL_CTYPE_BGR	Bleu-Green-Red
GFL_CTYPE_RGBA	Red-Green-Bleu-Alpha
GFL_CTYPE_ABGR	Alpha-Bleu-Green-Red

GFL\_CTYPE\_CMY

Cyan-Magenta-Yellow

GFL\_CTYPE\_CMYK

Cyan-Magenta-Yellow-Black

Write

Pointer to a write user function.

Tell

Pointer to a tell user function.

Seek

Pointer to a seek user function.

## See also

[gflGetDefaultSaveParams](#), [gflSaveBitmap](#), [gflSaveBitmapIntoHandle](#)

# GFL\_RECT

The **GFL\_RECT** structure define a rectangle.

```
typedef struct (  
    GFL_INT32 x,  
    GFL_INT32 y,  
    GFL_INT32 w,  
    GFL_INT32 h  
) GFL_RECT
```

## Members

x  
X position.

y  
Y position.

w  
Width.

h  
Height.

## See also

[gflCrop](#)

# GFL\_COLOR

The **GFL\_COLOR** structure allow to define a color.

```
typedef struct (  
    GFL_UINT16 Red,  
    GFL_UINT16 Green,  
    GFL_UINT16 Blue,  
    GFL_UINT16 Alpha  
} GFL_COLOR
```

## Members

Red

Define the red component.

Green

Define the green component.

Blue

Define the blue component.

Alpha

Define the alpha component.

## See also

[gflBalance](#), [gflResizeCanvas](#), [gflGetColorAt](#)

# GFL\_POINT

The **GFL\_POINT** structure allows to define a point.

```
typedef struct (  
    GFL_INT32 x,  
    GFL_INT32 y  
) GFL_POINT
```

## Members

- x  
X position.
- y  
Y position.

# GFL\_FILTER

The **GFL\_FILTER** structure allows to define a matrix for convolution (maximum 7x7).

```
typedef struct (  
    GFL_INT16 Size,  
    GFL_INT16 Matrix[7*7],  
    GFL_INT16 Divisor,  
    GFL_INT16 Bias  
) GFL_FILTER
```

## Members

Size

Define the width of the matrix (maximum 7).

Matrix

Define each values fo the matrix.

Divisor

Define the divisor to apply.

Bias

Define the bias to apply.

## Example

A "blur" matrix is defined like this:

Size = 3

Matrix = (1 2 1 2 4 2 1 2 1)

Divisor = 16

Bias = 0

## See also

[gflConvolve](#)

## Use with Visual Basic

The **GflLib** use with Visual Basic requires the use of modules. See examples for more informations, or contact us by e-mail.

### **GflLib.bas**

This file contains the API declarations, structures and constants required by **GflLib** and **GflLibs**.

### **GflLibExt.bas**

This file contains functions required to use **GflLib** with Visual Basic.

extGetGflBitmapFromPtr	Retrieves the data of a GFL_BITMAP structure from a pointer
extGetGflColorMapFromPtr	Retrieves the data of a GFL_COLORMAP structure from a pointer
extGetGflComments	Retrieves a dynamic array of string (comments) from a pointer
extGetStr	Retrieves a string from a pointer
extRTN	Trims and erases the NULL characters of a C string
extFarProc	Gets the pointer on a Visual Basic function with AddressOf
extShowBitmapOnDc	Display a GFL_BITMAP in a graphical context.

### **GflLibe.bas**

This file contains the API declarations, structures and constants required by **GflLibe**, extension of **GflLib**.

## GflLibeExt.bas

This file contains functions required to use **GflLibe** with Visual Basic.

extGetDIBFromPtr	Retrieve a DIB from a pointer
extShowBitmapOnDcEx	Display a GFL_BITMAP in a graphical context by using the ConvertBitmapIntoDIB function (GflLibe) (same as extShowBitmapIntoDC)
extShowTransparencyBitmapOnDcEx	Display a transparency GFL_BITMAP in a graphical context by using the ConvertBitmapIntoDIB function (GflLibe).

## GflReadData.bas

This file contains functions required to use callbacks in **LoadBitmapFromHandle** and **LoadPreviewFromHandle** with Visual Basic.

extLoadFile	Loads a file into a READ_DATA.
extSetDataToPtr	Copy the data of a READ_DATA structure in memory.
READ_ReadFunction	The function which reads the data.
READ_TellFunction	The function which gives the position of reading.
READ_SeekFunction	The function which sets the position.

## GflSaveData.bas

This file contains functions required to use callbacks in **SaveBitmapIntoHandle** with Visual Basic.

extSaveFile	Saves the data from a pointer on a SAVE_DATA structure in a file.
extWriteArray	Saves the data from a pointer in a array.
extGetDataToPtr	Copy in a SAVE_DATA structure the data in memory.
SAVE_ReadFunction	The function which writes the data.
SAVE_TellFunction	The function which gives the position of writing.
SAVE_SeekFunction	The function which sets the position.











# gflSaveBitmapIntoMemory

The **gflSaveBitmapIntoMemory** function save a picture in memory.

```
GFL_ERROR gflSaveBitmapIntoHandle(  
    GFL_UINT8 ** data,  
    GFL_UINT32 * data_length,  
    const GFL_BITMAP * bitmap,  
    GFL_SAVE_PARAMS * params  
);
```

## Parameters

data

Address of a pointer for data.

data\_length

Address of data's length.

handle

User handle. The Callbacks field of the

[GFL\\_SAVE\\_PARAMS](#) structure must be filled correctly.

bitmap

Address of a pointer to a [GFL\\_BITMAP](#) structure.

params

Pointer to a [GFL\\_SAVE\\_PARAMS](#) structure.

This structure must be filled correctly, in particular the FormatIndex field.

## Return value

The function returns GFL\_NO\_ERROR if it is successful or a value of [GFL\\_ERROR](#).

## See also

[gflGetDefaultLoadParams](#), [gflGetDefaultThumbnailParams](#), [gflGetDefaultSaveParams](#), [gflLoadBitmap](#),  
[gflLoadBitmapFromMemory](#), [gflLoadBitmapFromHandle](#), [gflLoadThumbnail](#), [gflLoadThumbnailFromMemory](#),  
[gflLoadThumbnailFromHandle](#), [gflSaveBitmapIntoMemory](#), [gflSaveBitmap](#)

# GFL\_EXIF\_DATA

The **GFL\_EXIF\_DATA** structure is used to get EXIF metadata.

```
typedef struct (  
    GFL_EXIF_ENTRY ItemsList[],  
    GFL_UINT32 NumberOfItems  
) GFL_EXIF_DATA
```

## Members

*ItemsList*

Address of a [GFL\\_EXIF\\_ENTRY](#) structure for the EXIF entries.

*NumberOfItems*

Number of EXIF informations.

## See also

[gflBitmapHasEXIF](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#)

# GFL\_IPTC\_DATA

The **GFL\_IPTC\_DATA** structure is used to get IPTC metadata.

```
typedef struct (  
    GFL_IPTC_ENTRY  ItemsList[],  
    GFL_UINT32  NumberOfItems  
} GFL_IPTC_DATA
```

## Members

*ItemsList*

Address of a [GFL\\_IPTC\\_ENTRY](#) structure for the IPTC entries.

*NumberOfItems*

Number of IPTC informations.

## See also

[gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflFreeIPTC](#)

# GFL\_EXIF\_ENTRY

The **GFL\_EXIF\_ENTRY** structure contains one EXIF tag.

```
typedef struct (  
    GFL_UINT32 Flag,  
    GFL_UINT32 Tag,  
    const char * Name,  
    const char * Value  
) GFL_EXIF_ENTRY
```

## Members

### Flag

Directory of the tag.

EXIF_MAIN_IFD	0	Main
EXIF_IFD_0	2	IFD 0 or Camera IFD
EXIF_INTEROPERABILITY_IFD	4	Interoperability IFD
EXIF_IFD_THUMBNAIL	8	Thumbnail IFD
EXIF_GPS_IFD	16	GPS IFD
EXIF_MAKERNOTE_IFD	32	Makernote IFD (Canon, Olympus, Minolta, Fuji, Nikon, Casio camera supported)

### Tag

Tag value.

### Name

Pointer to a null-terminated string that contains the label of the tag.

### Value

Pointer to a null-terminated string that contains the value of the tag.

## See also

[gflBitmapHasEXIF](#), [gflBitmapGetEXIF](#), [gflFreeEXIF](#), [GFL\\_EXIF\\_DATA](#)

# GFL\_IPTC\_ENTRY

The **GFL\_IPTC\_ENTRY** structure contains one IPTC tag.

```
typedef struct (  
    GFL_UINT32  Id,  
    const char * Name,  
    const char * Value  
} GFL_IPTC_ENTRY
```

## Members

*Id*

Tag ID.

*Name*

Pointer to a null-terminated string that contains the label of the tag.

*Value*

Pointer to a null-terminated string that contains the value of the tag.

## See also

[gflBitmapHasIPTC](#), [gflBitmapGetIPTC](#), [gflFreeIPTC](#), [GFL\\_IPTC\\_DATA](#)