BASS_FLAC_StreamCreateFile

Creates a sample stream from a FLAC file.

```
HSTREAM BASS_FLAC_StreamCreateFile(
    BOOL mem,
    void *file,
    QWORD offset,
    QWORD length,
    DWORD flags
);
```

Parameters

mem	TRUE = stream the file from memory.	
file	Filename (mem = FALSE) or a memory location (mem = TRUE).	
offset	File offset to begin streaming from (only used if mem = FALSE).	
length	Data length... 0 = use all data up to the end of the file (if mem = FALSE)	
flags	A combination of these flags.	
	BASS_SAMPLE_FLOAT	Use 32-bit floating-point sample data. Floating-point channels for info.
	BASS_SAMPLE_SOFTWARE	Force the stream to not use hardware mixing.
	BASS_SAMPLE_3D	Enable 3D functionality. This requires that the BASS_DEVICE_3D flag was specified when calling BASS_Init, and the stream must be mono. The SPEAK flags can not be used together with this flag.
	BASS_SAMPLE_LOOP	Loop the file. This flag can be toggled any time using BASS_ChannelFlags.
	BASS_SAMPLE_FX	Enable the old implementation of DirectX 8 effects. See the DX8 effect implementations section for details. Use BASS_ChannelSetFX to add effects to a stream.
	BASS_STREAM_AUTOFREE	Automatically free the stream when playback ends.
	BASS_STREAM_DECODE	Decode the sample data, without playing it. Use BASS_ChannelGetData to retrieve decoded sample data. The BASS_SAMPLE_3D, BASS_STREAM_AUTOFREE and SPEAKER flags can not be used together with this flag. The BASS_SAMPLE_SOFTWARE and BASS_SAMPLE_FX flags are also

	ignored.
BASS_SPEAKER_*xxx*	Speaker assignment flags. These flags have no effect when the stream is mor than stereo.
BASS_ASYNCFILE	Read the file asynchronously. When enabled, the file is read and buffered i parallel with the decoding, to reduce t chances of the decoder being affected I/O delays. This can be particularly us with slow storage media and/or low latency output. The size of the file buf is determined by the BASS_CONFIG_ASYNCFILE_BUF config option. This flag is ignored wh streaming from memory (*mem = TRU*
BASS_UNICODE	*file* is in UTF-16 form. Otherwise it is ANSI on Windows or Windows CE, a UTF-8 on other platforms.

Return value
If successful, the new stream's handle is returned, else 0 is returned. Use BASS_ErrorGetCode to get the error code.

Error codes

BASS_ERROR_INIT	BASS_Init has not been successfully called.
BASS_ERROR_NOTAVAIL	Only decoding channels (BASS_STREAM_DECODE) are allowed when using the "no sound" device. The BASS_STREAM_AUTOFREE flag is also unavailable to decoding channels.
BASS_ERROR_ILLPARAM	The *length* must be specified when streaming from memory.
BASS_ERROR_FILEOPEN	The file could not be opened.
BASS_ERROR_FILEFORM	The file's format is not recognised/supported.
BASS_ERROR_FORMAT	The sample format is not supported by the device/drivers. If the stream is more than stereo or the BASS_SAMPLE_FLOAT flag is used, it could be that they are not supported.
BASS_ERROR_SPEAKER	The specified SPEAKER flags are invalid. The device/drivers do not support them, they are attempting to assign a stereo stream to a mono speaker or 3D functionality is enabled.
BASS_ERROR_MEM	There is insufficent memory.
BASS_ERROR_NO3D	Could not initialize 3D support.
BASS_ERROR_UNKNOWN	Some other mystery problem!

Remarks
All FLAC sample resolutions from 8 to 32-bit are supported, but the output will be restricted to 16-bit unless the BASS_SAMPLE_FLOAT flag is used. The file's original resolution is available via BASS_ChannelGetInfo.

Use BASS_ChannelGetInfo to retrieve information on the format (sample rate, resolution, channels) of the stream. The playback length of the stream can be retrieved using BASS_ChannelGetLength. Until the whole file has been streamed, whatever length the file's header says is returned, which may or may not be exact.

FLAC streams have a few different types of tag available via BASS_ChannelGetTags. The FLAC format uses Ogg Vorbis tags, so the standard BASS_TAG_OGG and BASS_TAG_VENDOR tags apply for those. Embedded cuesheets are supported and are available via the BASS_TAG_FLAC_CUE tag, which gives a pointer to a TAG_FLAC_CUE structure. Embedded images are also supported and are available via the BASS_TAG_FLAC_PICTURE+<*index*> tag (*index=0* is the first picture), which gives a pointer to a TAG_FLAC_PICTURE structure. Application metadata blocks are also supported and available via the BASS_TAG_FLAC_METADATA+<*index*> tag (*index=0* is the first block), which gives a pointer to a TAG_FLAC_METADATA structure.

Chained Ogg FLAC files are supported, and a BASS_SYNC_OGG_CHANGE sync can be set via BASS_ChannelSetSync to be informed of when a new bitstream begins during decoding/playback of them, at which point new tags may be available. The length of a chained Ogg FLAC file will be unavailable until the entire file has been decoded, and seeking via BASS_ChannelSetPosition is not possible except for going back to the start (or the BASS_POS_DECODETO option is used).

To stream a file from the internet, use BASS_FLAC_StreamCreateURL. To stream from other locations, see BASS_FLAC_StreamCreateFileUser.

Platform-specific
Away from Windows, all mixing is done in software (by BASS), so the BASS_SAMPLE_SOFTWARE flag is unnecessary. The BASS_SAMPLE_FX flag is also ignored.

Example
Create a stream of a FLAC file.

```
HSTREAM stream=BASS_FLAC_StreamCreateFile(FALSE, "afile.flac", 0, 0
```

See also
BASS_FLAC_StreamCreateFileUser, BASS_FLAC_StreamCreateURL

BASS_ChannelGetInfo, BASS_ChannelGetLength, BASS_ChannelGetTags, BASS_ChannelPlay, BASS_ChannelSetAttribute, BASS_ChannelSetDSP, BASS_ChannelSetFX, BASS_ChannelSetLink, BASS_StreamFree, BASS_StreamGetFilePosition

BASS_FLAC_StreamCreateFileUser

Creates a sample stream from a FLAC file via user callback functions.

```
HSTREAM BASS_FLAC_StreamCreateFileUser(
    DWORD system,
    DWORD flags,
    BASS_FILEPROCS *procs,
    void *user
);
```

Parameters

system File system to use, one of the following.

STREAMFILE_NOBUFFER	Unbuffered.
STREAMFILE_BUFFER	Buffered.
STREAMFILE_BUFFERPUSH	Buffered, with the data pushed to BA via BASS_StreamPutFileData.

flags A combination of these flags.

BASS_SAMPLE_FLOAT	Use 32-bit floating-point sample data Floating-point channels for info.
BASS_SAMPLE_SOFTWARE	Force the stream to not use hardware mixing.
BASS_SAMPLE_3D	Enable 3D functionality. This require that the BASS_DEVICE_3D flag wa specified when calling BASS_Init, ar the stream must be mono. The SPEA flags can not be used together with th flag.
BASS_SAMPLE_LOOP	Loop the file. This flag can be toggle any time using BASS_ChannelFlags.
BASS_SAMPLE_FX	Enable the old implementation of Dir 8 effects. See the DX8 effect implementations section for details. U BASS_ChannelSetFX to add effects t stream.
BASS_STREAM_RESTRATE	Restrict the "download" rate of the fil the rate required to sustain playback. this flag is not used, then the file will downloaded as quickly as possible. T flag only has effect when using the STREAMFILE_BUFFER system.
BASS_STREAM_BLOCK	Download and play the file in smaller chunks. Uses a lot less memory than otherwise, but it's not possible to seek loop the stream; once it's ended, the f

		must be opened again to play it again This flag will automatically be applie when the file length is unknown. This also has the effect of restricting the download rate. This flag has no effec when using the STREAMFILE_NOBUFFER system
	BASS_STREAM_AUTOFREE	Automatically free the stream when playback ends.
	BASS_STREAM_DECODE	Decode the sample data, without play it. Use BASS_ChannelGetData to ret decoded sample data. The BASS_SAMPLE_3D, BASS_STREAM_AUTOFREE and SPEAKER flags can not be used toge with this flag. The BASS_SAMPLE_SOFTWARE and BASS_SAMPLE_FX flags are also ignored.
	BASS_SPEAKER_*xxx*	Speaker assignment flags. These flag have no effect when the stream is mo than stereo.
	BASS_ASYNCFILE	Read the file asynchronously. When enabled, the file is read and buffered parallel with the decoding, to reduce chances of the decoder being affected I/O delays. This can be particularly u with slow storage media and/or low latency output. The size of the file bu is determined by the BASS_CONFIG_ASYNCFILE_BUF config option. This flag only applies using the STREAMFILE_NOBUFFE system.
procs	The user defined file functions.	
user	User instance data to pass to the callback functions.	

Return value
If successful, the new stream's handle is returned, else 0 is returned. Use BASS_ErrorGetCode to get the error code.

Error codes

BASS_ERROR_INIT	BASS_Init has not been successfully called.
BASS_ERROR_NOTAVAIL	Only decoding channels (BASS_STREAM_DECODE) are allowed when using the "no sound" device. The BASS_STREAM_AUTOFREE flag is also unavailable to decoding channels.
BASS_ERROR_ILLPARAM	*system* is not valid.
BASS_ERROR_FILEFORM	The file's format is not recognised/supported.
BASS_ERROR_FORMAT	The sample format is not supported by the device/drivers. If the stream is more than stereo or the BASS_SAMPLE_FLOAT flag is used, it could be that they are not supported.
BASS_ERROR_SPEAKER	The specified SPEAKER flags are invalid. The device/drivers do not support them, they are attempting to assign a stereo stream to a mono speaker or 3D functionality is enabled.
BASS_ERROR_MEM	There is insufficent memory.
BASS_ERROR_NO3D	Could not initialize 3D support.
BASS_ERROR_UNKNOWN	Some other mystery problem!

Platform-specific
Away from Windows, all mixing is done in software (by BASS), so the BASS_SAMPLE_SOFTWARE flag is unnecessary. The BASS_SAMPLE_FX flag is also ignored.

See also
BASS_FLAC_StreamCreateFile, BASS_FLAC_StreamCreateURL

BASS_ChannelGetInfo, BASS_ChannelGetLength, BASS_ChannelGetTags, BASS_ChannelPlay, BASS_ChannelSetAttribute, BASS_ChannelSetDSP, BASS_ChannelSetFX, BASS_ChannelSetLink, BASS_StreamFree, BASS_FILEPROCS structure, BASS_CONFIG_NET_BUFFER

BASS_FLAC_StreamCreateURL

Creates a sample stream from an FLAC file on the internet, optionally receiving the downloaded data in a callback.

```
HSTREAM BASS_FLAC_StreamCreateURL(
    char *url,
    DWORD offset,
    DWORD flags,
    DOWNLOADPROC *proc,
    void *user
);
```

Parameters

url	URL of the file to stream. Should begin with "http://" or "https://" or "ftp://".	
offset	File position to start streaming from. This is ignored by some servers, specifically when the file length is unknown.	
flags	A combination of these flags.	
	BASS_SAMPLE_FLOAT	Use 32-bit floating-point sample data. See Floating-point channels for info.
	BASS_SAMPLE_SOFTWARE	Force the stream to not use hardware mixing.
	BASS_SAMPLE_3D	Enable 3D functionality. This requires that the BASS_DEVICE_3D flag was specified when calling BASS_Init, and the stream must be mono. The SPEAKER flags can not be used together with this flag.
	BASS_SAMPLE_LOOP	Loop the file. This flag can be toggled at any time using BASS_ChannelFlags. This flag is ignored when streaming in blocks (BASS_STREAM_BLOCK).
	BASS_SAMPLE_FX	Enable the old implementation of DirectX 8 effects. See the DX8 effect implementations section for details. Use BASS_ChannelSetFX to add effects to the stream.
	BASS_STREAM_RESTRATE	Restrict the download rate of the file to the rate required to sustain playback. If this flag is not used, then the file will be downloaded as quickly as the user's internet connection allows.
	BASS_STREAM_BLOCK	Download and play the file in smaller chunks. Uses a lot less

		memory than otherwise, but it's not possible to seek or loop the stream; once it's ended, the file must be opened again to play it again. This flag will automatically be applied when the file length is unknown, for example with Shout/Icecast streams. This flag also has the effect of resticting the download rate.
	BASS_STREAM_STATUS	Pass status info (HTTP/ICY tags) from the server to the DOWNLOADPROC callback during connection. This can be useful to determine the reason for a failure.
	BASS_STREAM_AUTOFREE	Automatically free the stream when playback ends.
	BASS_STREAM_DECODE	Decode the sample data, without playing it. Use BASS_ChannelGetData to retrieve decoded sample data. The BASS_SAMPLE_3D, BASS_STREAM_AUTOFREE and SPEAKER flags can not be used together with this flag. The BASS_SAMPLE_SOFTWARE and BASS_SAMPLE_FX flags are also ignored.
	BASS_SPEAKER_*xxx*	Speaker assignment flags. These flags have no effect when the stream is more than stereo.
	BASS_UNICODE	*url* is in UTF-16 form. Otherwise it is ANSI on Windows or Windows CE, and UTF-8 on other platforms.
proc	Callback function to receive the file as it is downloaded... NULL = no callback.	

user User instance data to pass to the callback function.

Return value
If successful, the new stream's handle is returned, else 0 is returned. Use BASS_ErrorGetCode to get the error code.

Error codes

BASS_ERROR_INIT	BASS_Init has not been successfully called.
BASS_ERROR_NOTAVAIL	Only decoding channels (BASS_STREAM_DECODE) are allowed when using the "no sound" device. The BASS_STREAM_AUTOFREE flag is also unavailable to decoding channels.
BASS_ERROR_NONET	No internet connection could be opened.
BASS_ERROR_ILLPARAM	*url* is not a valid URL.
BASS_ERROR_SSL	SSL/HTTPS support is not available.
BASS_ERROR_TIMEOUT	The server did not respond to the request within the timeout period, as set with the BASS_CONFIG_NET_TIMEOUT config option.
BASS_ERROR_FILEOPEN	The file could not be opened.
BASS_ERROR_FILEFORM	The file's format is not recognised/supported.
BASS_ERROR_FORMAT	The sample format is not supported by the device/drivers. If the stream is more than stereo or the BASS_SAMPLE_FLOAT flag is used, it could be that they are not supported (ie. no WDM drivers).
BASS_ERROR_SPEAKER	The specified SPEAKER flags are invalid. The device/drivers do not support them, they are attempting to assign a stereo stream to a mono speaker or 3D functionality is enabled.
BASS_ERROR_MEM	There is insufficent memory.
BASS_ERROR_NO3D	Could not initialize 3D support.
BASS_ERROR_UNKNOWN	Some other mystery problem!

Remarks
Use BASS_ChannelGetInfo to retrieve information on the format (sample rate, resolution, channels) of the stream. The playback length of the stream can be retrieved using BASS_ChannelGetLength. Until the whole file has been streamed, whatever length the file's header says is returned, which may or may not be exact.

FLAC streams have a few different types of tag available via BASS_ChannelGetTags. The FLAC format uses Ogg Vorbis tags, so the standard BASS_TAG_OGG and BASS_TAG_VENDOR tags apply. Embedded images are also supported and are available via the BASS_TAG_FLAC_PICTURE+<*index*> tag (*index=0* is the first picture), which gives a pointer to a TAG_FLAC_PICTURE structure. Application metadata blocks are also supported and available via the BASS_TAG_FLAC_METADATA+<*index*> tag (*index=0* is the first block), which gives a pointer to a TAG_FLAC_METADATA structure.

Chained Ogg FLAC streams are supported, and a BASS_SYNC_OGG_CHANGE sync can be set via BASS_ChannelSetSync to be informed of when a new bitstream begins during decoding/playback of them, at which point new tags may be available. FLAC does not have a constant or nominal bitrate; the BASS_ATTRIB_BITRATE attribute will give the average bitrate from the start to the current position.

When playing the stream, BASS will stall the playback if there is insufficient data to continue playing. Playback will automatically be resumed when sufficient data has been downloaded. BASS_ChannelIsActive can be used to check if the playback is stalled, and the progress of the file download can be checked with BASS_StreamGetFilePosition.

When streaming in blocks (BASS_STREAM_BLOCK flag), be careful not to stop/pause the stream for too long, otherwise the connection may timeout due to there being no activity and the stream will end prematurely.

When using an *offset*, the file length returned by BASS_StreamGetFilePosition can be used to check that it was successful by comparing it with the original file length. Another way to check is to inspect the HTTP headers retrieved with

BASS_ChannelGetTags.

Platform-specific
Away from Windows, all mixing is done in software (by BASS), so the BASS_SAMPLE_SOFTWARE flag is unnecessary. The BASS_SAMPLE_FX flag is also ignored.

See also
BASS_FLAC_StreamCreateFile, BASS_FLAC_StreamCreateFileUser

BASS_ChannelGetInfo, BASS_ChannelGetLength, BASS_ChannelGetTags, BASS_ChannelPlay, BASS_ChannelSetAttribute, BASS_ChannelSetDSP, BASS_ChannelSetFX, BASS_StreamFree, BASS_StreamGetFilePosition, DOWNLOADPROC callback, BASS_CONFIG_NET_AGENT, BASS_CONFIG_NET_BUFFER, BASS_CONFIG_NET_PREBUF, BASS_CONFIG_NET_PROXY, BASS_CONFIG_NET_TIMEOUT

Plugin system

As well as providing dedicated stream creation functions, BASSFLAC supports the BASS plugin system, adding FLAC file support to the standard BASS stream and sample creation functions: BASS_StreamCreateFile, BASS_StreamCreateURL, BASS_StreamCreateFileUser, and BASS_SampleLoad. This is enabled using the BASS_PluginLoad function.

TAG_FLAC_CUE structure

FLAC cuesheet tag structure.

```
typedef struct {
    char *catalog;
    DWORD leadin;
    BOOL iscd;
    DWORD ntracks;
    TAG_FLAC_CUE_TRACK *tracks;
} TAG_FLAC_CUE;
```

Members

catalog	Media catalog number.
leadin	The number of lead-in samples.
iscd	The cuesheet corresponds to a CD?
ntracks	The number of tracks.
tracks	The tracks.

Remarks

Further details can be found in the FLAC format specification, here:
flac.sourceforge.net/format.html

See also
BASS_ChannelGetTags

TAG_FLAC_CUE_TRACK structure

FLAC cuesheet tag track structure.

```
typedef struct {
    QWORD offset;
    DWORD number;
    char *isrc;
    DWORD flags;
    DWORD nindexes;
    TAG_FLAC_CUE_TRACK_INDEX *indexes;
} TAG_FLAC_CUE_TRACK;
```

Members

offset	Track offset in samples.	
number	The track number.	
isrc	International Standard Recording Code.	
flags	Any combination of the following flags.	
	TAG_FLAC_CUE_TRACK_DATA	Non-audio.
	TAG_FLAC_CUE_TRACK_PRE	Pre-emphasis.
nindexes	The number of indexes.	
indexes	The indexes.	

See also

TAG_FLAC_CUE structure

TAG_FLAC_CUE_TRACK_INDEX structure

FLAC cuesheet tag track index structure.

```
typedef struct {
    QWORD offset;
    DWORD number;
} TAG_FLAC_CUE_TRACK_INDEX;
```

Members

offset	Index offset in samples relative to the track offset.
number	The index number.

See also

TAG_FLAC_CUE_TRACK structure

TAG_FLAC_METADATA structure

FLAC application metadata tag structure.

```
typedef struct {
    char id[4];
    DWORD length;
    void *data;
} TAG_FLAC_METADATA;
```

Members

id	The application ID. A list of registered IDs is available at: www.xiph.org/flac/id.html
length	The size of *data* in bytes.
data	The metadata.

See also
BASS_ChannelGetTags

TAG_FLAC_PICTURE structure

FLAC picture tag structure.

```
typedef struct {
    DWORD apic;
    char *mime;
    char *desc;
    DWORD width;
    DWORD height;
    DWORD depth;
    DWORD colors;
    DWORD length;
    void *data;
} TAG_FLAC_PICTURE;
```

Members

apic	The picture type, according to the ID3v2 "APIC" frame specification: see www.id3.org/id3v2.3.0 for details.
mime	The MIME type. This may be "-->" to signify that *data* contains a URL of the picture rather than the picture data itself.
desc	A description of the picture, in UTF-8 form.
width	The width in pixels.
height	The height in pixels.
depth	The colour depth in bits-per-pixel.
colors	The number of colours used for indexed-colour pictures (eg. GIF).
length	The size of *data* in bytes.
data	The picture data.

Remarks

The *width*, *height*, *depth*, and *colors* members may be empty (0) so should not be depended on. That information can be obtained from the picture data itself.

See also
BASS_ChannelGetTags