FSBank Library API Reference

FSBank Library API Reference.

Functions, Structures, Enums
- Functions

FSBank_Build
FSBank_Close
FSBANK_DEBUGCALLBACK
FSBank_GetBuildCancel
FSBank_Init
FSBank_IsFormatAllowed
FSBank_SetBuildCancel
FSBank_SetDebugCallback
FSBank_SetUpdateCallback
FSBANK_UPDATALCALLBACK
- Structures

FSBANK_SAMPLE_DEFAULTS
- Enums

FSBANK_BUILDMODE
FSBANK_FORMAT
FSBANK_PLATFORM
FSBANK_RESULT
[API function]
FSBank_Build

Builds the fsb file(s).

```
FSBank_Built(fsbuild
  FSBANK_BUILDMODE buildmode,
  FSBANK_FORMAT format,
  FSBANK_PLATFORM platform,
  int basicheaders,
  const char *destfile_or_dir,
  int numsrcfiles,
  char **srcfile,
  FSBANK_SAMPLE_DEFAULTS **defaults,
  int dupdirstructure,
  const char *srcdir,
  int createincludes,
  int abortonerror
);
```

Parameters

- **buildmode**
  The type of FSB file(s) to generate. Either
  FSBANK_BUILDMODE_SINGLE,
  FSBANK_BUILDMODE_MULTI or
  FSBANK_BUILDMODE_INTERLEAVED.

- **platform**
  The platform to target. This may be necessary for platforms
  such as gamecube which require 32 byte padding.

- **format**
  The compression destination format. See FSBANK_FORMAT
  for a description of each format.
  If set to TRUE, this generates FSBs with small sample header
  data. They only contain basic information such as sample
  length, and everything else has its attributes inherited from the
  first sample (for example the default frequency). Set to 0 by
  default.
  Destination filename if using
  FSBANK_BUILDMODE_SINGLE or
**destfile_or_dir**  
FSBANK_BUILDMODE_INTERLEAVED. Destination directory if using FSBANK_BUILDMODE_MULTI.

**numsrcfiles**  
The number of source files to add to the FSB target.

**srcfile**  
Array of strings containing absolute paths to the source files to be added to the FSB target.

**defaults**  
Array of FSBANK_SAMPLE_DEFAULTS structures that align with the src file string array. This is used to specify defaults for the files being built so they load that way in FMOD for playback. Can be NULL. See remarks if NULL.

For format type FSBANK_BUILDMODE_MULTI only. This will create the same source directory structure to the destination target directory. FALSE as default.

**dupdirstructure**  
Only if dupdirstructure = TRUE. The root of the source directory to be recreated in the destination directory. NULL as default if dupdirstructure is not specified.

**srcdir**  
Create C header files with

**createincludes**  
If set to TRUE, compilation will be aborted if an error occurs (i.e. file not found), all output/intermediate files will be deleted and FSBANK_ERR_COMPILATION_ABORTED will be returned

**abortonerror**

---

**Return Value**

On success, FSBANK_OK is returned.
On failure, an FSBANK_RESULT error code is returned.

**Remarks**

This is a blocking function. You may want to execute this function from a thread to allow it to compile in the background (ie to update a GUI).
'basicheaders' are useful to save memory in a game when an FSB has been generated with thousands of samples in it.
Basicheaders does not refer to the C header file generated, it refers to the data stored inside the FSB.
If defaults are not specified for samples, for PlayStation 2 VAG, XBox XADPCM and Gamecube XADPCM, samples(streams) are automatically set to
FSOUND_HW3D if mono, FSOUND_HW2D if stereo.

See Also

FSBANK_BUILDMODE, FSBANK_FORMAT, FSBANK_PLATFORM, FSBANK_RESULT, FSBANK_SAMPLE_DEFAULTS, FSBank_SetBuildCancel
[API function]
FSBank_Close

Shuts down the FSBank Library.

\[
\text{FSBANK\_RESULT FSBank\_Close();}
\]

**Return Value**

On success, FSBANK_OK is returned.
On failure, an FSBANK\_RESULT error code is returned.

**Remarks**

**See Also**

FSBank_Init, FSBANK\_RESULT
[API function]
**FSBank_GetBuildCancel**

Gets the cancel state of the FSB compiler. Set by FSBank_SetBuildCancel.

```c
FSBANK_RESULT FSBank_GetBuildCancel(
    int *cancel
);
```

**Parameters**

*cancel* Pointer to an integer to return either TRUE (non zero) or FALSE (zero) for the state of the build process if it has been flagged to cancel or not.

**Return Value**

On success, FSBANK_OK is returned.
On failure, an FSBANK_RESULT error code is returned.

**Remarks**

**See Also**

FSBANK_RESULT, FSBank_SetBuildCancel
**FSBank_Init**

Initializes FSBank library. This must be called first.

```c
FSBANK_RESULT FSBank_Init();
```

**Return Value**

On success, FSBANK_OK is returned.  
On failure, an FSBANK_RESULT error code is returned.

**Remarks**

To get each platform to work, certain files or settings have to be present.  
For PlayStation 2, ENCVA
G.DLL has to be present.  
For GameCube, DSP
TOOL.DLL has to be present.  
For XBox, the XBox XDK has to be installed to work.

**See Also**

[FSBank_Close](#), [FSBANK_RESULT](#)
[API function]
**FSBank_IsFormatAllowed**

Returns whether the format specified can be targetted or not.

```c
FSBANK_RESULT FSBank_IsFormatAllowed( FSBANK_FORMAT format );
```

**Parameters**

`format` The format to query.

**Return Value**

On success, FSBANK_OK is returned. On failure, an FSBANK_RESULT error code is returned.

**Remarks**

The return value is based on whether the correct dll or sdk is installed properly. For example.

For FSBANK_FORMAT_VAG, ENCVAG.DLL has to be present. This is usually only shipped with the Sony Libraries and you have to be a registered sony developer to have this.

For FSBANK_FORMAT_GCADPCM, DSPTOOL.DLL has to be present. This is usually only shipped with the NDK and you have to be a registered Nintendo developer to have this.

For FSBANK_FORMAT_XADPCM, the XBox XDK has to be installed to work.

**See Also**

[FSBANK_FORMAT](#) , [FSBANK_RESULT](#)
FSBank_SetBuildCancel

Sets or clears the 'cancel' flag inside the compiler, so that compilation can be halted mid compile.

FSBANK_RESULT FSBank_SetBuildCancel(
    int cancel
);

Parameters

cancel Should be either TRUE (non zero) or FALSE (zero).

Return Value

On success, FSBANK_OK is returned.
On failure, an FSBANK_RESULT error code is returned.

Remarks

Because the FSBank_Build function is a blocking function, you will not get a chance to call this during compilation, but if you call FSBank_Build from another thread, then this can be called during the compilation process.
Another alternative is to call it from an FSBank_UpdateCallback which will give a change to terminate mid compile without needing to execute FSBank_Build from a separate thread, but will have less accuracy (ie it will have to wait until a file has been compressed before cancelling).

See Also

FSBank_Build , FSBank_GetBuildCancel , FSBANK_RESULT
[API function]
**FSBank_SetDebugCallback**

Sets a callback for whenever a debug message is logged through FSBank library.

```
FSBANK_RESULT FSBank_SetDebugCallback(
    FSBANK_DEBUGCALLBACK callback,
    void *userdata
);
```

**Parameters**

- `callback` Pointer to function to receive callback.
- `userdata` Pointer to user data that will be passed back to the callback when it is issued.

**Return Value**

On success, FSBANK_OK is returned.
On failure, an FSBANK_RESULT error code is returned.

**Remarks**

Debug messages are generated during the build process, and can be used for logging purposes. The user can display them or write them to a file, or whatever they feel like.

**See Also**

[FSBANK_DEBUGCALLBACK](#), [FSBANK_RESULT](#)
[API function]
FSBank_SetUpdateCallback

Sets a callback for whenever a file has been compiled into an FSB.

```c
FSBANKRESULT FSBank_SetUpdateCallback( 
    FSBANK_UPDATECALLBACK callback, 
    void *userdata
);
```

**Parameters**

- `callback` Pointer to function to receive callback.
- `userdata` Pointer to user data that will be passed back to the callback when it is issued.

**Return Value**

On success, FSBANK_OK is returned. On failure, an FSBANK_RESULT error code is returned.

**Remarks**

**See Also**

- `FSBANK_RESULT`, `FSBANK_UPDATECALLBACK`
FSBANK_BUILDMODE

Describes the target build type or method of creating the FSB file(s).

Enumerators

FSBANK_BUILDMODE_SINGLE

// This creates a single FSB file with multiple sounds in it, or a standard sound bank.

FSBANK_BUILDMODE_MULTI

// This creates multiple FSB files with 1 sound in each. The destfile_or_dir parameter of FSBank_Build is then interpreted as a directory and not a file.

FSBANK_BUILDMODE_INTERLEAVED

// This creates a single FSB file with a single sound in it, but with all the source files interleaved/multiplexed into it so that when it is played, all files play at once, and are given a channel each.

See Also

FSBank_Build

This document copyright ©Firelight Technologies, Pty, Ltd, 1999-2003. All rights reserved.
Generated Wed Apr 07 15:29:01 2004 by SourceDoc v0.10, the automated source code documenter.
FSBANK_FORMAT

Describes the target format.

Enumerators

FSBANK_FORMAT_VAG // VAG (SPU2) (3.5:1) PlayStation 2 Only. Hardware decompression, no cpu hit.
FSBANK_FORMAT_GCADPCM // GCADPCM (3.5:1) GameCube Only. Hardware decompression, no cpu hit.
FSBANK_FORMAT_XADPCM // XADPCM (3.5:1) XBox only. Hardware decompression, no cpu hit.
FSBANK_FORMAT_PCM // PCM (1:1) All Platforms.
FSBANK_FORMAT_SOURCE // Retain original format. All platforms (except PlayStation 2 unless using pcm wav files).
FSBANK_FORMAT_IMAADPCM // IMA ADPCM (3.5:1) All platforms except PlayStation 2.
FSBANK_FORMAT_MAX

See Also

FSBank_Build
FSBANK_PLATFORM

Describes the target platform.

Enumerators

FSBANK_PLATFORM_PS2 // Sony PlayStation 2
FSBANK_PLATFORM_GC // Nintendo GameCube
FSBANK_PLATFORM_XBOX // Microsoft XBox
FSBANK_PLATFORM_CROSS // Cross platform. Only PCM is truly supported on all platforms.

FSBANK_PLATFORM_MAX

See Also

FSBank_Builder
[Enum]
FSBANK_RESULT

Errorcode returned by all FSBank commands

Enumerators

FSBANK_OK
FSBANK_ERR_INIT // Failed to initialize
FSBANK_ERR_UNINITIALIZED // FSBank_Init hasn't been called yet.
FSBANK_ERR_FILE_DIRNOTFILE // The target is an existing file. The specified build mode requires a destination directory, not a file.
FSBANK_ERR_FILE_DESTFILE // Cannot create destination file. File may be in use or read only
FSBANK_ERR_FILE_WORKING // Cannot create working file. File may be in use or read only
FSBANK_ERR_FILE_HEADER // Cannot create destination c header file. File may be in use or read only
FSBANK_ERR_FILE_EOF // End of file was encountered unexpectedly.
FSBANK_ERR_FILE_OS // An operating system based file error was encountered. Could cause corruption or failure of FSB to be created.
FSBANK_ERR_INVALID_PARAM // An invalid parameter was passed to this function
FSBANK_ERR_INVALID_FORMAT // A dll was missing for this format or the environment wasn't set up properly.
FSBANK_ERR_CANCELLED // The build process was cancelled during compilation by the user.
FSBANK_ERR_COMPILATION_ABORTED // Compilation aborted due to error
See Also

FSBank_Init
[API function]
FSBANK_DEBUGCALLBACK

Debug callback. This is called whenever a debug message is generated by the FSBank library.

```c
void __stdcall FSBANK_DEBUGCALLBACK(
    const char *debugstring,
    void *userdata
);
```

Parameters

- `debugstring` Debug string generated by the action executed in the FSBank library.
- `userdata` User data specified by FSBank_SetDebugCallback

Return Value

`void`

Remarks

See Also

[FSBank_SetDebugCallback](#)
[API function]
FSBANK_UPDATECALLBACK

Update callback. This is called whenever a source file is compiled into an FSB.

void __stdcall FSBANK_UPDATECALLBACK(
    int index,
    int memused,
    void *userdata
);

Parameters

index The index of the source file in the FSB being compiled.
memused Compressed sound memory used so far in the FSB.
userdata User data specified by FSBank_SetUpdateCallback

Return Value

void

Remarks

This can be used for progress bars or updating of the interface. This is usually called before and after each source file is compiled to allow the interface to get the before and after values which may be necessary if it is a multi file FSB batch build, ie FSBANK_BUILDMODE_MULTI.

See Also

FSBANK_BUILDMODE, FSBank_SetUpdateCallback
FSBANK_SAMPLE_DEFAULTS

Structure containing default values for various sample attributes.

Members

float  mindistance // Minimum volume distance in "units"
float  maxdistance // Maximum volume distance in "units"
int    deffreq    // Sample default speed in hz
int    defvol     // Sample default volume
int    defpan     // Sample default pan
int    defpri     // Sample priority. 0 = low priority, 255=high priority
int    varfreq    // Frequency variation in hz
int    varvol     // Volume variation
int    varpan     // Pan variation
unsigned mode     // FSOUND_MODES bits. Bits allowed are
                  FSOUND_LOOP_NORMAL, FSOUND_LOOP_BIDI,
                  FSOUND_2D, FSOUND_HW2D and FSOUND_HW3D

See Also

FSBank_Build

Referenced By

FSBank_Build