Table of Contents

Fix Me! This page is not fully translated, yet. Please help completing the translation.
(remove this paragraph once the translation is finished)

SDK API > BS2_SetDebugFileLogEx

BS2_SetDebugFileLogEx

[+ 2.8.3] Log messages within the SDK can be output as a split file and used for debugging applications.
The fourth factor, fileMaxSizeMB, allows you to specify a maximum size per file, in MB.
If you set this value to 0, you will not proceed with file segmentation saving.
The file can be specified as either an absolute path or a relative path, and log messages that occur during subsequent SDK operations are automatically generated in that path. The file name stored at this time is YYYYMMDD_x.Take the form of log (_x is given sequentially from 0 whenever fileMaxSize is exceeded)

Declaration

```
#include "BS_API.h"

int BS2_SetDebugFileLogEx(uint32_t level, uint32_t module, const char*
logPath, int fileMaxSizeMB);
```

Parameter

- [In] *level* : Specify error level

Macro	Value	Description
DEBUG_LOG_FATAL	0x00000001	Fatal Error
DEBUG_LOG_ERROR	0x00000002	General Error
DEBUG_LOG_WARN	0x00000004	Warning
DEBUG_LOG_API	0x00000008	API Calls IN and OUT
DEBUG_LOG_INFO	0x00000010	Information other than errors
DEBUG_LOG_TRACE	0x00000100	SDK Self-Debugging Information
DEBUG_LOG_SYSTEM	0x0000000F	Output all error levels and general information
DEBUG_LOG_OPERATION_ALL	0x000000FF	Output all error levels
DEBUG_LOG_ALL	0xFFFFFFFF	All Information Output

- [In] *module* : Specify a module

Macro	Value	
DEBUG_MODULE_KEEP_ALIVE	0x00000001	Keep alive module
DEBUG_MODULE_SOCKET_MANAGER	0x00000002	Socket Management Module
DEBUG_MODULE_SOCKET_HANDLER	0x00000004	Socket Operation Module
DEBUG_MODULE_DEVICE	0x00000008	Device Module
DEBUG_MODULE_DEVICE_MANAGER	0x00000010	Device Management Module
DEBUG_MODULE_EVENT_DISPATCHER	0x00000020	Event Handler Module
DEBUG_MODULE_API	0x00000040	API
DEBUG_MODULE_MISC	0x000080	Other
DEBUG_MODULE_PACKET	0x00000100	Communication Packet Processing Module
DEBUG_MODULE_NOTIFY_MANAGER	0x00000400	Notify Processing Module
DEBUG_MODULE_EVENT	0x00000800	USB event log processing module
DEBUG_MODULE_USB	0x00001000	USB import processing module
DEBUG_MODULE_ALL	0xFFFFFFFF	Full module

- [In] *logPath* : Path to which the log file will be output

Return Value

If successfully done, BS_SDK_SUCCESS will be returned.
If there is an error, the corresponding error code will be returned.

Sample Code(C++)

sample_setdebugfilelogex.cpp

```
const char* CURRENT_DIR = ".";
const int MAX_SIZE_LOG_FILE = 100;  // 100MB
int sdkResult = BS2_SetDebugFileLogEx(DEBUG_LOG_ALL, DEBUG_MODULE_ALL,
CURRENT_DIR, MAX_SIZE_LOG_FILE);
if (BS_SDK_SUCCESS != sdkResult)
{
    printf("BS2_SetDebugFileLogEx call failed: %d", sdkResult);
    return;
}
```

Sample Code (C#)

sample_setdebugfilelogex.cs

```
const string CURRENT_DIR = ".";
const int MAX_SIZE_LOG_FILE = 100;  // 100MB
IntPtr ptrDir = Marshal.StringToHGlobalAnsi(CURRENT_DIR);
result =
(BS2ErrorCode)API.BS2_SetDebugFileLogEx(Constants.DEBUG_LOG_OPERATION_A
LL, Constants.DEBUG_MODULE_ALL, ptrDir, MAX_SIZE_LOG_FILE);
Marshal.FreeHGlobal(ptrDir);
if (result != BS2ErrorCode.BS_SDK_SUCCESS)
{
    Console.WriteLine("Got error({0}).", result);
    return;
}
```