| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [Common/src](http://docs.google.com/index.html) - ErrorHandler.cpp (source / [functions](http://docs.google.com/ErrorHandler.cpp.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 32 | 36 | 88.9 % | | Date: | 2012-03-23 |  | Functions: | 10 | 11 | 90.9 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 13 | 18 | 72.2 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : : #include <stdarg.h>  2 : :   3 : : #include "ErrorHandler.h"  4 : : #include "ThreadManager.h"  5 : :   6 : : namespace nft {  7 : :   8 : : ErrorHandler \*  9 : 425 : ErrorHandler::instance() {  10 [ + + ][ + - ]: 425 : static ErrorHandler pObjErrorHandler;  11 : 425 : return &pObjErrorHandler;  12 : : };  13 : :   14 : 73 : ErrorHandler::ErrorHandler(){  15 : 73 : mErrType = DPL::FATAL\_ERROR;  16 : 73 : mErrLevel = ERROR\_LEVEL\_NORMAL;  17 : 73 : mErrCode = 0;  18 : : }  19 : :   20 : 73 : ErrorHandler::~ErrorHandler(){  21 : : }  22 : :   23 : : //Construct the error message from the error table and the parameters passed at call  24 : : const std::string  25 : : ErrorHandler::getErrorMsg(const ErrorHandler::ErrorID eiCode, ...)  26 : 425 : {  27 : 425 : va\_list va\_args;  28 : 425 : va\_start(va\_args, eiCode);  29 : 425 : char buf[ BUFSIZ ];  30 : 425 : vsprintf(buf, ErrorHandler::mMsgTable[eiCode].mMsg.c\_str(), va\_args);  31 : 425 : return std::string(buf);  32 : : }  33 : :   34 : : // Create the error object and throw it to the function catching it  35 : : void  36 : : ErrorHandler::errorAdd(DPL::Error \*inObjSubError,  37 : : const char \* inFileName, const char \* inSourceModuleName,  38 : : int32\_t inLineNum, const ErrorID inErrorCode,  39 : : const std::string &inErrMessage,  40 : 425 : ThreadManager \*inThreadManager) throw(DPL::Error \*) {  41 [ + + ]: 425 : if(NULL == inObjSubError)  42 : : {  43 : 110 : mErrType = ErrorHandler::mMsgTable[inErrorCode].mErrType;  44 : 110 : mErrLevel = ErrorHandler::mMsgTable[inErrorCode].mErrLevel;  45 : 425 : mErrCode = inErrorCode;  46 : : }  47 : :   48 : : DPL::Error \*e = new DPL::Error(getErrorType(inErrorCode), inErrorCode,  49 : : const\_cast<char \*>(inFileName),  50 : : const\_cast<char \*>(inSourceModuleName), inLineNum,  51 : 425 : const\_cast<char \*>(inErrMessage.c\_str()), inObjSubError);  52 [ + + ]: 425 : if( !inObjSubError ) //Only for leaf level error  53 : : {  54 : 110 : mErrType = getErrorType(inErrorCode);  55 : 110 : mErrLevel = getErrorLevel(inErrorCode);  56 : 110 : mErrCode = inErrorCode;  57 : : }  58 [ + + ]: 425 : if( 0 != inThreadManager )  59 : : {  60 : 69 : inThreadManager->handleException( e );  61 : 69 : return;  62 : : }  63 [ # # ]: 425 : throw (e);  64 : : }  65 : :   66 : : void  67 : 0 : ErrorHandler::setErrorStatus(unsigned int ErrCode) {  68 : 0 : mErrType = ErrorHandler::mMsgTable[ErrCode].mErrType;  69 : 0 : mErrLevel = ErrorHandler::mMsgTable[ErrCode].mErrLevel;  70 : 0 : mErrCode = ErrCode;  71 : : }  72 : :   73 : : //Table of error messages  74 [ + + ]: 80190 : const ErrorHandler::ErrorMsg ErrorHandler::mMsgTable[eErrNumElements]=  75 : : {  76 : : //General  77 : : { eErrMessageString, "%s", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  78 : : { eErrFunction, "Error in child function %s", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  79 : : { eErrRecordIndex, "Invalid relation record index in VSB %d", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  80 : :   81 : : //System related  82 : : { eErrUnhandled, "Unhandled exception", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  83 : : { eErrSignalHandler, "Caught by signal handler with signal %d", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  84 : : { eErrSystem, "System Error", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  85 : : { eErrMemoryAllocation, "Failed to allocate memory: %s", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  86 : : { eErrMmap, "Error while mapping the file %s onto memory", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  87 : : { eErrMunmap, "Error while un-mapping the file %s from memory", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},   88 : : { eErrUnload, "Error while un-loading the file %s%s from memory", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},   89 : : { eErrThread, "Error in threading function", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  90 : : { eErrFileOpen, "Failure in opening %s ", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  91 : : { eErrFileNotFound, "File %s not found ", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  92 : : { eErrDataMode, "Incorrect data mode ", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  93 : : { eErrTruncate, "Failed to allocate file of given size %u", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  94 : :   95 : : //Library level  96 : : { eErrIndexOutOfBounds, "Index %lu is out of bounds. Valid range: %lu-%lu", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  97 : : { eErrRangeOverflow, "Value %ld crosses the allowed range. Valid range: %ld-%ld", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  98 : :   99 : : //Application level  100 : : { eErrCreateMemoryWindow, "Error in creating the shared or remote memory window of size %u", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  101 : : { eErrMessageSend, "Error while sending message to DPM from process %s", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  102 : : { eErrReadData, "Error in reading data from remote/shared memory in process %s", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  103 : : { eErrLoadData, "Error in loading data", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  104 : : { eErrProcessManagerInit, "Failure in initializing process manager in process %s ", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  105 : : { eErrCmlParser, "Error parsing command line: %s", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  106 : : { eErrParse, "Parser Error", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  107 : : { eErrReadParam, "Error while reading the parameters from DPM", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  108 : :   109 : : //Processing engine  110 : : { eErrInit, "Error in initializing process %s ", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  111 : : { eErrExecute, "Error in executing process %s ", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  112 : : { eErrGMCValueBeyondLimit, "Error : GMC value (%f, %f) at position (%ld, %ld) beyond limit value (%f)", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  113 : : { eErrCellSizeMoreThanSF, "Cell span to be covered (%d,%d) is more than SF size (%d,%d)", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  114 : :   115 : :   116 : : { eErrConfParse, "Error : %s is not found", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL},  117 : : { eErrConfVal, "Error : value of %s is incorrect, expected : %s", DPL::FATAL\_ERROR, ERROR\_LEVEL\_FATAL}  118 [ + + ][ # # ]: 41310 : };  119 : :   120 : :   121 : 41310 : } // namespace nft |

|  |
| --- |
| Generated by: [LCOV version 1.9](http://ltp.sourceforge.net/coverage/lcov.php) |