| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [InputParams/src](http://docs.google.com/index.html) - ConverterParams.cpp (source / [functions](http://docs.google.com/ConverterParams.cpp.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 115 | 155 | 74.2 % | | Date: | 2012-03-23 |  | Functions: | 31 | 33 | 93.9 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 35 | 118 | 29.7 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : : #include "MemInfo.h"  2 : : #include "AITable.h"  3 : : #include "ConverterCommandLineOptions.h"  4 : : #include "ConverterParams.h"  5 : : #include "DIVTable.h"  6 : : #include "DoseTable.h"  7 : : #include "DOCTable.h"  8 : : #include "GMCMap.h"  9 : : #include "KBRMap.h"  10 : : #include "KBRTable.h"  11 : : #include "SetlTbl.h"  12 : : #include "SlopeData.h"  13 : :   14 : : namespace nft {  15 : :   16 : 2430 : ConverterParams::Cleaner::~Cleaner() {  17 : 1215 : ConverterParams::instance()->memCleanup();  18 : : }  19 : :   20 : : ConverterParams::ConverterParams()  21 : 1215 : : mStartupMessage(),  22 : : mLayerParams(),  23 : : mColumnParams(),  24 : : mCommandLine(),  25 : : mAITable(),  26 : : mDIVTable(),  27 : : mDOCTable(),  28 : : mGMCMap(),  29 : : mKBRMap(),  30 : : mKBRTable(),  31 : : mDoseTable(),  32 : : mSlopeData(),  33 : : mMainSetlTbl(),  34 : : mSubSetlTbl(),  35 : : mTDSetlTbl(),  36 [ # # ][ # # ]: 313470 : mSizerSetlTbl() {  [ # # ][ # # ]  [ # # ][ # # ]  [ + - ][ + + ]  [ # # ][ # # ]  [ # # ][ # # ]  [ # # ][ # # ]  [ # # ][ # # ]  37 : : }  38 : :   39 : :   40 : 1215 : ConverterParams::~ConverterParams() {  41 : 1215 : delete mStartupMessage;  42 : 1215 : mStartupMessage = 0;  43 : 1215 : delete mHeader;  44 : 1215 : mHeader = 0;  45 : : }  46 : :   47 : : ConverterParams \*  48 : 1215 : ConverterParams::instance() {  49 [ + + ][ + - ]: 757019 : static ConverterParams obj;  [ + + ][ + - ]  [ # # ][ # # ]  50 : 755804 : return &obj;  51 : : }  52 : :   53 : : void  54 : : ConverterParams::setStartupMessage(  55 : 1119 : DPMANAGER\_MESSAGE\_CONVERTER \*inMsgConverter) {  56 : 1119 : mStartupMessage = inMsgConverter;  57 : : }  58 : :   59 : : void  60 : : ConverterParams::setHeader(  61 : 1119 : DPMANAGER\_MESSAGE\_HEADER \*inMsgHeader) {  62 : 1119 : mHeader = inMsgHeader;  63 : : }  64 : :   65 : : void  66 : : ConverterParams::setLayerParams(  67 : : const DPMANAGER\_MESSAGE\_SFL\_INFO \*inLayerInfo,  68 : 1119 : size\_t inNumParams) {  69 : 1119 : mLayerParams = new DPMANAGER\_MESSAGE\_SFL\_INFO[ inNumParams ];  70 : 1119 : ::memcpy( mLayerParams, inLayerInfo,  71 : : inNumParams \* sizeof( DPMANAGER\_MESSAGE\_SFL\_INFO ) );  72 : : }  73 : :   74 : : void  75 : : ConverterParams::setColumnParams(  76 : 1119 : const sgs\_column\_t &inColumnParams ) {  77 : 1119 : mColumnParams = inColumnParams;  78 : : }  79 : :   80 : : void  81 : : ConverterParams::setCommandLineOptions(  82 : 0 : ConverterCommandLineOptions \*inCmdlOpts ) {  83 : 0 : mCommandLine = inCmdlOpts;  84 : : }  85 : :   86 : : void  87 : : ConverterParams::setAITable( const MemInfo &inMemory,  88 : 26 : bool isSharedMemory, DPL::ExternalMemory \*inExternalMem) throw(DPL::Error \*){  89 : 26 : try{  90 : 26 : mAITable = AITable::create( inMemory,  91 : : isSharedMemory, inExternalMem );  92 : : }  93 [ # # ]: 0 : catch( DPL::Error \*pChild )  94 : : {  95 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  96 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  97 : : ErrorHandler::eErrFunction,  98 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  99 : : "ConverterParams::setAITable"));  100 [ # # ]: 26 : return;  101 : : }  102 : : }  103 : :   104 : : void  105 : : ConverterParams::setDIVTable( const MemInfo &inMemory,  106 : 1098 : bool isSharedMemory, DPL::ExternalMemory \*inExternalMem) throw(DPL::Error \*){  107 : 1098 : try{  108 : 1098 : mDIVTable = DIVTable::create( inMemory,  109 : : isSharedMemory, inExternalMem );  110 : : }  111 [ # # ]: 0 : catch( DPL::Error \*pChild )  112 : : {  113 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  114 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  115 : : ErrorHandler::eErrFunction,  116 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  117 : : "ConverterParams::setDIVTable"));  118 [ # # ]: 1098 : return;  119 : : }  120 : : }  121 : :   122 : : void  123 : : ConverterParams::setGMCMap( const MemInfo &inMemory,  124 : : uint32\_t inRows, uint32\_t inCols, bool inIsSharedMemory,  125 : 35 : DPL::ExternalMemory \*inExternalMemory ) throw(DPL::Error \*){  126 : 35 : try{  127 : 35 : mGMCMap = GMCMap::create( inMemory, inRows, inCols,  128 : : inIsSharedMemory, inExternalMemory );  129 : : }  130 [ # # ]: 0 : catch( DPL::Error \*pChild )  131 : : {  132 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  133 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  134 : : ErrorHandler::eErrFunction,  135 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  136 : : "ConverterParams::setGMCMap"));  137 [ # # ]: 35 : return;  138 : : }  139 : : }  140 : :   141 : : void  142 : : ConverterParams::setDOCTable(const MemInfo &inMemory,  143 : 1095 : bool isSharedMemory, DPL::ExternalMemory \*inExternalMem) throw(DPL::Error \*){  144 : 1095 : try{  145 : 1095 : mDOCTable = DOCTable::create( inMemory,  146 : : isSharedMemory, inExternalMem);  147 : : }  148 [ # # ]: 0 : catch( DPL::Error \*pChild )  149 : : {  150 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  151 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  152 : : ErrorHandler::eErrFunction,  153 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  154 : : "ConverterParams::setDOCTable"));  155 [ # # ]: 1095 : return;  156 : : }  157 : : }  158 : :   159 : : void  160 : : ConverterParams::setKBRMap( const MemInfo &inMemory,  161 : : uint32\_t inRows, uint32\_t inCols, bool inIsSharedMemory,  162 : 1089 : DPL::ExternalMemory \*inExternalMemory ) throw(DPL::Error \*) {  163 : :   164 : 1089 : try{  165 : 1089 : mKBRMap = KBRMap::create( inMemory, inRows, inCols,  166 : : inIsSharedMemory, inExternalMemory );  167 : : }  168 [ # # ]: 0 : catch( DPL::Error \*pChild )  169 : : {  170 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  171 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  172 : : ErrorHandler::eErrFunction,  173 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  174 : : "ConverterParams::setKBRMap"));  175 [ # # ]: 1089 : return;  176 : : }  177 : :   178 : : }  179 : :   180 : : void  181 : : ConverterParams::setKBRTable( const MemInfo &inMemory,  182 : : uint32\_t inRows, uint32\_t inCols, bool inIsSharedMemory,  183 : 1086 : DPL::ExternalMemory \*inExternalMemory ) throw(DPL::Error \*){  184 : 1086 : try{  185 : 1086 : mKBRTable = KBRTable::create( inMemory, inRows, inCols,  186 : : inIsSharedMemory, inExternalMemory );  187 : : }  188 [ # # ]: 0 : catch( DPL::Error \*pChild )  189 : : {  190 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  191 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  192 : : ErrorHandler::eErrFunction,  193 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  194 : : "ConverterParams::setKBRTable"));  195 [ # # ]: 1086 : return;  196 : : }  197 : :   198 : : }  199 : :   200 : : void  201 : : ConverterParams::setDoseTable(const MemInfo &inMemory,  202 : 200 : bool isSharedMemory, DPL::ExternalMemory \*inExternalMem) throw(DPL::Error \*){  203 : 200 : try{  204 : 200 : mDoseTable = DoseTable::create( inMemory,  205 : : isSharedMemory, inExternalMem);  206 : : }  207 [ # # ]: 0 : catch( DPL::Error \*pChild )  208 : : {  209 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  210 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  211 : : ErrorHandler::eErrFunction,  212 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  213 : : "ConverterParams::setDoseTable"));  214 [ # # ]: 200 : return;  215 : : }  216 : : }  217 : :   218 : : void  219 : : ConverterParams::setSlopeData(const MemInfo &inMemory,  220 : 197 : bool isSharedMemory, DPL::ExternalMemory \*inExternalMem) throw(DPL::Error \*){  221 : 197 : try{  222 : 197 : mSlopeData = SlopeData::create( inMemory,  223 : : isSharedMemory, inExternalMem);  224 : : }  225 [ # # ]: 0 : catch( DPL::Error \*pChild )  226 : : {  227 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  228 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  229 : : ErrorHandler::eErrFunction,  230 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  231 : : "ConverterParams::setSlopeData"));  232 [ # # ]: 197 : return;  233 : : }  234 : : }  235 : :   236 : : void  237 : : ConverterParams::setMainSetlTbl( const MemInfo &inMemory,  238 : : uint32\_t inRows, uint32\_t inCols, bool inIsSharedMemory,  239 : 1094 : DPL::ExternalMemory \*inExternalMemory ) throw(DPL::Error \*){  240 : 1094 : try{  241 : 1094 : mMainSetlTbl = SetlTbl::create( inMemory, inRows, inCols,  242 : : inIsSharedMemory, inExternalMemory );  243 : : }  244 [ # # ]: 0 : catch( DPL::Error \*pChild )  245 : : {  246 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  247 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  248 : : ErrorHandler::eErrFunction,  249 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  250 : : "ConverterParams::setMainSetlTbl"));  251 [ # # ]: 1094 : return;  252 : : }  253 : : }  254 : :   255 : : void  256 : : ConverterParams::setSubSetlTbl( const MemInfo &inMemory,  257 : : uint32\_t inRows, uint32\_t inCols, bool inIsSharedMemory,  258 : 1094 : DPL::ExternalMemory \*inExternalMemory )throw(DPL::Error \*) {  259 : 1094 : try{  260 : 1094 : mSubSetlTbl = SetlTbl::create( inMemory, inRows, inCols,  261 : : inIsSharedMemory, inExternalMemory );  262 : : }  263 [ # # ]: 0 : catch( DPL::Error \*pChild )  264 : : {  265 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  266 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  267 : : ErrorHandler::eErrFunction,  268 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  269 : : "ConverterParams::setSubSetlTbl"));  270 [ # # ]: 1094 : return;  271 : : }  272 : : }  273 : :   274 : : void  275 : : ConverterParams::setTDSetlTbl( const MemInfo &inMemory,  276 : : uint32\_t inRows, uint32\_t inCols, bool inIsSharedMemory,  277 : 1094 : DPL::ExternalMemory \*inExternalMemory ) throw(DPL::Error \*){  278 : 1094 : try{  279 : 1094 : mTDSetlTbl = SetlTbl::create( inMemory, inRows, inCols,  280 : : inIsSharedMemory, inExternalMemory );  281 : : }  282 [ # # ]: 0 : catch( DPL::Error \*pChild )  283 : : {  284 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  285 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  286 : : ErrorHandler::eErrFunction,  287 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  288 : : "ConverterParams::setTDSetlTbl"));  289 [ # # ]: 1094 : return;  290 : : }  291 : : }  292 : :   293 : : void  294 : : ConverterParams::setSizerSetlTbl( const MemInfo &inMemory,  295 : : uint32\_t inRows, uint32\_t inCols, bool inIsSharedMemory,  296 : 1094 : DPL::ExternalMemory \*inExternalMemory ) throw(DPL::Error \*){  297 : 1094 : try{  298 : 1094 : mSizerSetlTbl = SetlTbl::create( inMemory, inRows, inCols,  299 : : inIsSharedMemory, inExternalMemory );  300 : : }  301 [ # # ]: 0 : catch( DPL::Error \*pChild )  302 : : {  303 : 0 : ErrorHandler \*pErr = ErrorHandler::instance();  304 : 0 : pErr->errorAdd(pChild, \_\_FILE\_\_, \_\_FUNCTION\_\_, \_\_LINE\_\_,  305 : : ErrorHandler::eErrFunction,  306 : : pErr->getErrorMsg(ErrorHandler::eErrFunction,  307 : : "ConverterParams::setSizerSetlTbl"));  308 [ # # ]: 1094 : return;  309 : : }  310 : : }  311 : :   312 : : void  313 : 1119 : ConverterParams::setRefData( const MemInfo &inRefData ) {  314 : 1119 : mRefData = inRefData;  315 : : }  316 : :   317 : : void  318 : 1119 : ConverterParams::setLinkData( const MemInfo &inLinkData ) {  319 : 1119 : mLinkData = inLinkData;  320 : : }  321 : :   322 : : void  323 : 1119 : ConverterParams::setCellData( const MemInfo &inCellData ) {  324 : 1119 : mCellData = inCellData;  325 : : }  326 : :   327 : : void  328 : 0 : ConverterParams::setCellCommonData( const MemInfo &inCellCommonData ) {  329 : 0 : mCellCommonData = inCellCommonData;  330 : : }  331 : :   332 : : void  333 : 1261 : ConverterParams::setVSBAU( uint64\_t inVsbAU ) {  334 : 1261 : mVSBAU = inVsbAU;  335 : : }  336 : :   337 : : void  338 : 1215 : ConverterParams::memCleanup() {  339 [ + + ]: 1215 : delete [] mLayerParams;  340 : 1215 : mLayerParams = 0;  341 : : //delete mCommandLine;  342 [ + + ]: 1215 : delete mAITable;  343 : 1215 : mAITable = 0;  344 [ + + ]: 1215 : delete mDIVTable;  345 : 1215 : mDIVTable = 0;  346 [ + + ]: 1215 : delete mDOCTable;  347 : 1215 : mDOCTable = 0;  348 [ + + ]: 1215 : delete mGMCMap;  349 : 1215 : mGMCMap = 0;  350 [ + + ]: 1215 : delete mKBRMap;  351 : 1215 : mKBRMap = 0;  352 [ + + ]: 1215 : delete mKBRTable;  353 : 1215 : mKBRTable = 0;  354 [ + + ]: 1215 : delete mDoseTable;  355 : 1215 : mDoseTable = 0;  356 [ + + ]: 1215 : delete mSlopeData;  357 : 1215 : mSlopeData = 0;  358 [ + + ]: 1215 : delete mMainSetlTbl;  359 : 1215 : mMainSetlTbl = 0;  360 [ + + ]: 1215 : delete mSubSetlTbl;  361 : 1215 : mSubSetlTbl = 0;  362 [ + + ]: 1215 : delete mTDSetlTbl;  363 : 1215 : mTDSetlTbl = 0;  364 [ + + ]: 1215 : delete mSizerSetlTbl;  365 : 1215 : mSizerSetlTbl = 0;  366 : 1215 : delete [](mRefData.getMem());  367 : 1215 : mRefData.setMem(0);  368 : 1215 : mRefData.setMemSize(0);  369 : 1215 : delete [](mCellData.getMem());  370 : 1215 : mCellData.setMem(0);  371 : 1215 : mCellData.setMemSize(0);  372 : 1215 : delete [](mLinkData.getMem());  373 : 1215 : mLinkData.setMem(0);  374 : 1215 : mLinkData.setMemSize(0);  375 : : }  376 : :   377 : 1215 : } // namespace nft |

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