| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [Common/include](http://docs.google.com/index.html) - CellQueue.h (source / [functions](http://docs.google.com/CellQueue.h.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 13 | 13 | 100.0 % | | Date: | 2012-03-23 |  | Functions: | 5 | 5 | 100.0 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 0 | 0 | - | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : : #ifndef NFT\_CELLQUEUE\_H  2 : : #define NFT\_CELLQUEUE\_H  3 : :   4 : : #include <list>  5 : :   6 : : namespace nft {  7 : :   8 : : template<class \_CellType>  9 : : class CellQueue {  10 : : public:  11 : : typedef CellQueue \* Pointer;  12 : : typedef \_CellType CellType;  13 : : typedef typename CellType::Pointer CellPointer;  14 : :   15 : : CellQueue();  16 : :   17 : : ~CellQueue();  18 : :   19 : : CellQueue(const CellQueue<\_CellType> & inSource);  20 : :   21 : : CellQueue<\_CellType> &  22 : : operator=(const CellQueue<\_CellType> & inSource);  23 : :   24 : : void  25 : : scheduleCell(CellPointer inCell);  26 : :   27 : : //Return NULL if no more cell is presnt  28 : : CellPointer  29 : : getNextCell();  30 : :   31 : : bool  32 : : isEmpty() const;  33 : :   34 : : private:  35 : : std::list<CellPointer> mCells;  36 : :   37 : : };  38 : : template<class \_CellType>  39 : : CellQueue<\_CellType>::CellQueue()  40 : 138 : :mCells() {  41 : : }  42 : :   43 : : template<class \_CellType>  44 : 92 : CellQueue<\_CellType>::~CellQueue() {  45 : 46 : mCells.empty();  46 : : }  47 : :   48 : : template<class \_CellType>  49 : : CellQueue<\_CellType>::CellQueue(  50 : : const CellQueue<\_CellType> & inSource)  51 : : :mCells( inSource.mCells ) {  52 : : }  53 : :   54 : : template<class \_CellType>  55 : : CellQueue<\_CellType> &  56 : : CellQueue<\_CellType>::operator=(  57 : : const CellQueue<\_CellType> & inSource) {  58 : : if( &inSource != this )  59 : : {  60 : : mCells = inSource.mCells;  61 : : }  62 : : return \*this;  63 : : }  64 : :   65 : : template<class \_CellType>  66 : : void  67 : : CellQueue<\_CellType>::scheduleCell(  68 : 63 : CellQueue::CellPointer inCell) {  69 : 63 : mCells.push\_back( inCell );  70 : : }  71 : :   72 : : //Return NULL if no more cell is presnt  73 : : template<class \_CellType>  74 : : typename CellQueue<\_CellType>::CellPointer  75 : 63 : CellQueue<\_CellType>::getNextCell() {  76 : 63 : CellPointer cell = 0;  77 : 63 : if( !mCells.empty() )  78 : : {  79 : 63 : cell = \*(mCells.begin());  80 : 63 : mCells.pop\_front();  81 : : }  82 : 63 : return cell;  83 : : }  84 : :   85 : : template<class \_CellType>  86 : : bool  87 : 235 : CellQueue<\_CellType>::isEmpty() const {  88 : 235 : return mCells.empty();  89 : : }  90 : :   91 : :   92 : : } // namespace nft  93 : : #endif |

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