| LCOV - code coverage report |
| --- |
|  |
| | Current view: | [top level](http://docs.google.com/index.html) - [Common/include](http://docs.google.com/index.html) - Position.h (source / [functions](http://docs.google.com/Position.h.func.html)) |  |  | Hit | Total | Coverage | | --- | --- | --- | --- | --- | --- | --- | | Test: | EBM9K Converter Coverage Report |  | Lines: | 37 | 38 | 97.4 % | | Date: | 2012-03-23 |  | Functions: | 55 | 57 | 96.5 % | | Legend: | Lines: hit not hit | Branches: + taken - not taken # not executed |  | Branches: | 30 | 54 | 55.6 % | |  | |
|  |

|  |
| --- |
| Branch data Line data Source code  1 : : #ifndef NFT\_POSITION\_H  2 : : #define NFT\_POSITION\_H  3 : :   4 : :   5 : : #include <stdint.h>  6 : :   7 : : namespace nft {  8 : :   9 : : template<typename \_Tp>  10 : : class Position {  11 : : public:  12 : : inline const \_Tp getX() const;  13 : :   14 : : void  15 : : setX(\_Tp value);  16 : :   17 : : inline const \_Tp getY() const;  18 : :   19 : : void  20 : : setY(\_Tp value);  21 : :   22 : :   23 : : Position(\_Tp inX = 0, \_Tp inY = 0);  24 : :   25 : :   26 : : ~Position();  27 : :   28 : :   29 : : Position(const Position<\_Tp> & source);  30 : :   31 : : template<typename \_Up>  32 : : Position(const Position<\_Up> & source);  33 : :   34 : : Position<\_Tp> &  35 : : operator =(const Position<\_Tp> & source);  36 : :   37 : : template<typename \_Up>  38 : : Position<\_Tp> &  39 : : operator =(const Position<\_Up> & source);  40 : :   41 : : bool  42 : : operator== (const Position<\_Tp>& rhs) const;  43 : :   44 : : bool  45 : : operator < (const Position<\_Tp>& rhs) const;  46 : :   47 : : Position<\_Tp>   48 : : operator +(const Position<\_Tp> & source) const ;  49 : :   50 : : Position<\_Tp>   51 : : operator +=(const Position<\_Tp> & source);  52 : :   53 : : private:  54 : : \_Tp mX;  55 : :   56 : : \_Tp mY;  57 : :   58 : : };  59 : : template<typename \_Tp>  60 [ + # ][ - # ]: 270405 : inline const \_Tp Position<\_Tp>::getX() const {  [ + # # ]  [ - + ][ + ]  [ + - ][ + - ]  [ + + ][ + + ]  [ + - ][ - + ]  [ # # ]  61 : : return mX;  62 : : }  63 : :   64 : : template<typename \_Tp>  65 : : void  66 : 89065 : Position<\_Tp>::setX(\_Tp value) {  67 : 89065 : mX = value;  68 : : }  69 : :   70 : : template<typename \_Tp>  71 [ + - ][ + + ]: 301484 : inline const \_Tp Position<\_Tp>::getY() const {  [ + + ][ + - ]  [ + - ][ - + ]  [ + - ][ - + ]  [ # # ]  72 : : return mY;  73 : : }  74 : :   75 : : template<typename \_Tp>  76 : : void  77 : 92168 : Position<\_Tp>::setY(\_Tp value) {  78 : 92168 : mY = value;  79 : : }  80 : :   81 : : template<typename \_Tp>  82 : 388 :   83 : : Position<\_Tp>::Position(\_Tp inX, \_Tp inY)   84 : 330125 : : mX(inX),   85 : 330125 : mY(inY){  86 : : }  87 : :   88 : : template<typename \_Tp>  89 : :   90 : 587354 : Position<\_Tp>::~Position() {  91 : : }  92 : :   93 : : template<typename \_Tp>  94 : : template<typename \_Up>  95 : : Position<\_Tp>::Position(const Position<\_Up> & source)   96 : 18705 : : mX(source.getX()),  97 : 37410 : mY(source.getY()){  98 : : }  99 : :   100 : 3200 : template<typename \_Tp>  101 : : Position<\_Tp>::Position(const Position<\_Tp> & source)   102 : 238640 : : mX(source.mX),  103 : 238640 : mY(source.mY){  104 : : }  105 : :   106 : 472 : template<typename \_Tp>  107 : : template<typename \_Up>  108 : : Position<\_Tp> &  109 : 67 : Position<\_Tp>::operator=(const Position<\_Up> & source) {  110 : 67 : mX = source.getX();  111 : 67 : mY = source.getY();  112 : 67 : return (\*this);  113 : : }  114 : :   115 : : template<typename \_Tp>  116 : : Position<\_Tp> &  117 : 166620 : Position<\_Tp>::operator=(const Position<\_Tp> & source) {  118 [ + - ][ + - ]: 166620 : if(this != &source)  119 : : {  120 : 166620 : mX = source.mX;  121 : 166620 : mY = source.mY;  122 : : }  123 : 166620 : return (\*this);  124 : : }  125 : :   126 : : template<typename \_Tp>  127 : : bool   128 : : Position<\_Tp>::operator==(const Position<\_Tp> &rhs) const  129 : 464 : {  130 [ + - ][ + - ]: 464 : return (mX == rhs.mX && mY == rhs.mY);  131 : : }  132 : :   133 : : template<typename \_Tp>  134 : : bool   135 : : Position<\_Tp>::operator < (const Position<\_Tp> &rhs) const  136 : 15913 : {  137 [ + + ][ + + ]: 15913 : return (mX < rhs.mX || mY < rhs.mY);  138 : : }  139 : :   140 : : template<typename \_Tp>  141 : : Position<\_Tp>   142 : : Position<\_Tp>::operator + ( const Position<\_Tp> & rhs) const  143 : 1261 : {  144 : 1261 : return Position<\_Tp>( (mX + rhs.mX), (mY + rhs.mY) );  145 : : }  146 : :   147 : : template<typename \_Tp>  148 : : Position<\_Tp>   149 : : Position<\_Tp>::operator += ( const Position<\_Tp> & rhs)  150 : 1309 : {  151 : 1309 : mX += rhs.mX;  152 : 1309 : mY += rhs.mY;  153 : 1309 : return \*this;  154 : : }  155 : :   156 : :   157 : : typedef Position<int8\_t> BytePosition;  158 : : typedef Position<uint8\_t> UBytePosition;  159 : : typedef Position<int64\_t> LongPosition;  160 : : typedef Position<uint64\_t> ULongPosition;  161 : : typedef Position<int16\_t> ShortPosition;  162 : : typedef Position<int16\_t> UShortPosition;  163 : : typedef Position<int32\_t> IntPosition;  164 : : typedef Position<uint32\_t> UIntPosition;  165 : : typedef Position<double> DoublePosition;  166 : : typedef UIntPosition InputPatternSize;  167 : : typedef UShortPosition PatternSize;  168 : : typedef IntPosition PatternPosition;  169 : :   170 : : } // namespace nft  171 : : #endif |

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