Contents

1 MAMDA C++ API Namespace Index 1
  1.1 MAMDA C++ API Namespace List 1

2 MAMDA C++ API Hierarchical Index 3
  2.1 MAMDA C++ API Class Hierarchy 3

3 MAMDA C++ API Class Index 9
  3.1 MAMDA C++ API Class List 9

4 MAMDA C++ API File Index 19
  4.1 MAMDA C++ API File List 19

5 MAMDA C++ API Namespace Documentation 23
  5.1 Wombat Namespace Reference 23

6 MAMDA C++ API Class Documentation 59
  6.1 Wombat::char_str_less_than Struct Reference 59
  6.2 Wombat::MamdaAndQuery Class Reference 60
  6.3 Wombat::MamdaAuctionFields Class Reference 61
  6.4 Wombat::MamdaAuctionHandler Class Reference 63
  6.5 Wombat::MamdaAuctionListener Class Reference 66
  6.6 Wombat::MamdaAuctionRecap Class Reference 77
  6.7 Wombat::MamdaAuctionUpdate Class Reference 81
  6.8 Wombat::MamdaBasicErrorListener Class Reference 85
  6.9 Wombat::MamdaBasicEvent Class Reference 86
6.10 Wombat::MamdaBasicMsgListener Class Reference .......................... 97
6.11 Wombat::MamdaBasicQualityListener Class Reference ..................... 98
6.12 Wombat::MamdaBasicRecap Class Reference .................................. 99
6.13 Wombat::MamdaBasicSubscription Class Reference .......................... 106
6.14 Wombat::MamdaBookAtomicBookHandler Class Reference .................. 110
6.15 Wombat::MamdaBookAtomicGap Class Reference ............................. 113
6.16 Wombat::MamdaBookAtomicLevel Class Reference ............................ 115
6.17 Wombat::MamdaBookAtomicLevelEntry Class Reference ....................... 121
6.18 Wombat::MamdaBookAtomicLevelEntryHandler Class Reference ............. 126
6.19 Wombat::MamdaBookAtomicLevelHandler Class Reference .................... 128
6.20 Wombat::MamdaBookAtomicListener Class Reference ........................ 130
6.21 Wombat::MamdaCheckerHandler Class Reference .............................. 146
6.22 Wombat::MamdaCommonFields Class Reference ............................... 148
6.23 Wombat::MamdaConcreteBasicEvent Class Reference .......................... 153
6.24 Wombat::MamdaContainsAllQuery Class Reference ......................... 162
6.25 Wombat::MamdaContainsQuery Class Reference ............................... 164
6.26 Wombat::MamdaCurrencyFields Class Reference .............................. 166
6.27 Wombat::MamdaCurrencyHandler Class Reference ............................. 168
6.28 Wombat::MamdaCurrencyListener Class Reference ........................... 171
6.29 Wombat::MamdaCurrencyRecap Class Reference ............................... 180
6.30 Wombat::MamdaCurrencyUpdate Class Reference .............................. 183
6.31 Wombat::MamdaDataException Class Reference ................................ 186
6.32 Wombat::MamdaDateQuery Class Reference .................................... 187
6.33 Wombat::MamdaEqualsQuery Class Reference .................................. 188
6.34 Wombat::MamdaErrorListener Class Reference ................................. 190
6.35 Wombat::MamdaFields Class Reference ........................................ 191
6.36 Wombat::MamdaFundamentalFields Class Reference .......................... 192
6.37 Wombat::MamdaFundamentalHandler Class Reference ........................ 197
6.38 Wombat::MamdaFundamentalListener Class Reference ........................ 199
6.39 Wombat::MamdaFundamentals Class Reference ................................. 212
6.40 Wombat::MamdaLock Class Reference ............................................ 220
<table>
<thead>
<tr>
<th>Section</th>
<th>Reference</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.72</td>
<td>Wombat::MamdaOrderBook::bidEntryIterator Class Reference</td>
<td>373</td>
</tr>
<tr>
<td>6.73</td>
<td>Wombat::MamdaOrderBook::bidIterator Class Reference</td>
<td>377</td>
</tr>
<tr>
<td>6.74</td>
<td>Wombat::MamdaOrderBookBasicDelta Class Reference</td>
<td>381</td>
</tr>
<tr>
<td>6.75</td>
<td>Wombat::MamdaOrderBookBasicDeltaList Class Reference</td>
<td>388</td>
</tr>
<tr>
<td>6.76</td>
<td>Wombat::MamdaOrderBookBasicDeltaList::iterator Class Reference</td>
<td>394</td>
</tr>
<tr>
<td>6.77</td>
<td>Wombat::MamdaOrderBookChecker Class Reference</td>
<td>398</td>
</tr>
<tr>
<td>6.78</td>
<td>Wombat::MamdaOrderBookCheckerHandler Class Reference</td>
<td>401</td>
</tr>
<tr>
<td>6.79</td>
<td>Wombat::MamdaOrderBookClear Class Reference</td>
<td>403</td>
</tr>
<tr>
<td>6.80</td>
<td>Wombat::MamdaOrderBookComplexDelta Class Reference</td>
<td>405</td>
</tr>
<tr>
<td>6.81</td>
<td>Wombat::MamdaOrderBookConcreteComplexDelta Class Reference</td>
<td>406</td>
</tr>
<tr>
<td>6.82</td>
<td>Wombat::MamdaOrderBookConcreteSimpleDelta Class Reference</td>
<td>415</td>
</tr>
<tr>
<td>6.83</td>
<td>Wombat::MamdaOrderBookDelta Class Reference</td>
<td>424</td>
</tr>
<tr>
<td>6.84</td>
<td>MamdaOrderBookDepthFilter Class Reference</td>
<td>426</td>
</tr>
<tr>
<td>6.85</td>
<td>Wombat::MamdaOrderBookDuplicateEntry Class Reference</td>
<td>432</td>
</tr>
<tr>
<td>6.86</td>
<td>Wombat::MamdaOrderBookEntry Class Reference</td>
<td>434</td>
</tr>
<tr>
<td>6.87</td>
<td>Wombat::MamdaOrderBookEntryFilter Class Reference</td>
<td>448</td>
</tr>
<tr>
<td>6.88</td>
<td>Wombat::MamdaOrderBookEntryManager Class Reference</td>
<td>449</td>
</tr>
<tr>
<td>6.89</td>
<td>Wombat::MamdaOrderBookException Class Reference</td>
<td>452</td>
</tr>
<tr>
<td>6.90</td>
<td>Wombat::MamdaOrderBookFields Class Reference</td>
<td>454</td>
</tr>
<tr>
<td>6.91</td>
<td>Wombat::MamdaOrderBookGap Class Reference</td>
<td>461</td>
</tr>
<tr>
<td>6.92</td>
<td>Wombat::MamdaOrderBookHandler Class Reference</td>
<td>463</td>
</tr>
<tr>
<td>6.93</td>
<td>Wombat::MamdaOrderBookInvalidEntry Class Reference</td>
<td>468</td>
</tr>
<tr>
<td>6.94</td>
<td>Wombat::MamdaOrderBookListener Class Reference</td>
<td>470</td>
</tr>
<tr>
<td>6.95</td>
<td>Wombat::MamdaOrderBookMissingEntry Class Reference</td>
<td>485</td>
</tr>
<tr>
<td>6.96</td>
<td>Wombat::MamdaOrderBookPriceLevel Class Reference</td>
<td>487</td>
</tr>
<tr>
<td>6.97</td>
<td>Wombat::MamdaOrderBookPriceLevel::iterator Class Reference</td>
<td>506</td>
</tr>
<tr>
<td>6.98</td>
<td>Wombat::MamdaOrderBookRecap Class Reference</td>
<td>510</td>
</tr>
<tr>
<td>6.99</td>
<td>Wombat::MamdaOrderBookSimpleDelta Class Reference</td>
<td>512</td>
</tr>
<tr>
<td>6.100</td>
<td>Wombat::MamdaOrderBookTypes Class Reference</td>
<td>514</td>
</tr>
<tr>
<td>6.101</td>
<td>Wombat::MamdaOrderImbalanceFields Class Reference</td>
<td>518</td>
</tr>
<tr>
<td>6.102</td>
<td>Wombat::MamdaOrderImbalanceHandler Class Reference</td>
<td>523</td>
</tr>
</tbody>
</table>
## CONTENTS

6.134 Wombat::MamdaSymbolSourceHandler Class Reference ........................ 794
6.135 Wombat::MamdaTradeCancelOrError Class Reference ......................... 795
6.136 Wombat::MamdaTradeChecker Class Reference .................................. 807
6.137 Wombat::MamdaTradeClosing Class Reference .................................... 810
6.138 Wombat::MamdaTradeCorrection Class Reference .............................. 812
6.139 Wombat::MamdaTradeFields Class Reference ..................................... 828
6.140 Wombat::MamdaTradeGap Class Reference ........................................ 834
6.141 Wombat::MamdaTradeHandler Class Reference ................................... 837
6.142 Wombat::MamdaTradeListener Class Reference ................................. 842
6.143 Wombat::MamdaTradeOutOfSequence Class Reference ......................... 906
6.144 Wombat::MamdaTradePossiblyDuplicate Class Reference ..................... 915
6.145 Wombat::MamdaTradeRecap Class Reference ..................................... 924
6.146 Wombat::MamdaTradeReport Class Reference .................................... 957
6.147 map Class Reference ................................................................. 969

7 MAMDA C++ API File Documentation .................................................. 971
7.1 MamdaAuctionFields.h File Reference ............................................. 971
7.2 MamdaAuctionHandler.h File Reference ........................................... 972
7.3 MamdaAuctionListener.h File Reference ........................................... 973
7.4 MamdaAuctionRecap.h File Reference .............................................. 974
7.5 MamdaAuctionUpdate.h File Reference ............................................. 975
7.6 MamdaBasicEvent.h File Reference ................................................ 976
7.7 MamdaBasicRecap.h File Reference ................................................ 977
7.8 MamdaBasicSubscription.h File Reference ...................................... 978
7.9 MamdaBookAtomicBookHandler.h File Reference .................. ........................ 979
7.10 MamdaBookAtomicGap.h File Reference ........................................ 980
7.11 MamdaBookAtomicLevel.h File Reference ...................................... 981
7.12 MamdaBookAtomicLevelEntry.h File Reference ................................. 982
7.13 MamdaBookAtomicLevelEntryHandler.h File Reference ....................... 983
7.14 MamdaBookAtomicLevelHandler.h File Reference ............................. 984
7.15 MamdaBookAtomicListener.h File Reference .................................. 985

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
<table>
<thead>
<tr>
<th>Section</th>
<th>File Reference</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.16</td>
<td>MамдаCheckerHandler.h</td>
<td>986</td>
</tr>
<tr>
<td>7.17</td>
<td>MамдаCheckerType.h</td>
<td>987</td>
</tr>
<tr>
<td>7.18</td>
<td>MамдаCommonFields.h</td>
<td>988</td>
</tr>
<tr>
<td>7.19</td>
<td>MамдаConcreteBasicEvent.h</td>
<td>989</td>
</tr>
<tr>
<td>7.20</td>
<td>MамдаConfig.h</td>
<td>990</td>
</tr>
<tr>
<td>7.21</td>
<td>MамдаCurrencyFields.h</td>
<td>991</td>
</tr>
<tr>
<td>7.22</td>
<td>MамдаCurrencyHandler.h</td>
<td>992</td>
</tr>
<tr>
<td>7.23</td>
<td>MамдаCurrencyListener.h</td>
<td>993</td>
</tr>
<tr>
<td>7.24</td>
<td>MамдаCurrencyRecap.h</td>
<td>994</td>
</tr>
<tr>
<td>7.25</td>
<td>MамдаCurrencyUpdate.h</td>
<td>995</td>
</tr>
<tr>
<td>7.26</td>
<td>MамдаDataException.h</td>
<td>996</td>
</tr>
<tr>
<td>7.27</td>
<td>MамдаErrorListener.h</td>
<td>997</td>
</tr>
<tr>
<td>7.28</td>
<td>MамдаFields.h</td>
<td>998</td>
</tr>
<tr>
<td>7.29</td>
<td>MамдаFieldState.h</td>
<td>999</td>
</tr>
<tr>
<td>7.30</td>
<td>MамдаFundamentalFields.h</td>
<td>1000</td>
</tr>
<tr>
<td>7.31</td>
<td>MамдаFundamentalHandler.h</td>
<td>1001</td>
</tr>
<tr>
<td>7.32</td>
<td>MамдаFundamentalListener.h</td>
<td>1002</td>
</tr>
<tr>
<td>7.33</td>
<td>MамдаFundamentals.h</td>
<td>1003</td>
</tr>
<tr>
<td>7.34</td>
<td>MамдаLock.h</td>
<td>1004</td>
</tr>
<tr>
<td>7.35</td>
<td>MамдаMsgListener.h</td>
<td>1006</td>
</tr>
<tr>
<td>7.36</td>
<td>MамдаMultiParticipantHandler.h</td>
<td>1007</td>
</tr>
<tr>
<td>7.37</td>
<td>MамдаMultiParticipantManager.h</td>
<td>1008</td>
</tr>
<tr>
<td>7.38</td>
<td>MамдаMultiSecurityHandler.h</td>
<td>1009</td>
</tr>
<tr>
<td>7.39</td>
<td>MамдаMultiSecurityManager.h</td>
<td>1010</td>
</tr>
<tr>
<td>7.40</td>
<td>MамдаNewsFields.h</td>
<td>1011</td>
</tr>
<tr>
<td>7.41</td>
<td>MамдаNewsHeadline.h</td>
<td>1012</td>
</tr>
<tr>
<td>7.42</td>
<td>MамдаNewsHeadlineHandler.h</td>
<td>1013</td>
</tr>
<tr>
<td>7.43</td>
<td>MамдаNewsManager.h</td>
<td>1014</td>
</tr>
<tr>
<td>7.44</td>
<td>MамдаNewsMetaData.h</td>
<td>1015</td>
</tr>
<tr>
<td>7.45</td>
<td>MамдаNewsQuery.h</td>
<td>1016</td>
</tr>
<tr>
<td>7.46</td>
<td>MамдаNewsQueryHandler.h</td>
<td>1017</td>
</tr>
<tr>
<td>7.47 MamdaNewsStory.h File Reference</td>
<td>1018</td>
<td></td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>7.48 MamdaNewsStoryHandler.h File Reference</td>
<td>1019</td>
<td></td>
</tr>
<tr>
<td>7.49 MamdaNewsTypes.h File Reference</td>
<td>1020</td>
<td></td>
</tr>
<tr>
<td>7.50 MamdaOptionalConfig.h File Reference</td>
<td>1021</td>
<td></td>
</tr>
<tr>
<td>7.51 MamdaOptionAtTheMoneyCompareType.h File Reference</td>
<td>1022</td>
<td></td>
</tr>
<tr>
<td>7.52 MamdaOptionChain.h File Reference</td>
<td>1023</td>
<td></td>
</tr>
<tr>
<td>7.53 MamdaOptionChainHandler.h File Reference</td>
<td>1024</td>
<td></td>
</tr>
<tr>
<td>7.54 MamdaOptionChainListener.h File Reference</td>
<td>1025</td>
<td></td>
</tr>
<tr>
<td>7.55 MamdaOptionChainView.h File Reference</td>
<td>1026</td>
<td></td>
</tr>
<tr>
<td>7.56 MamdaOptionChainViewRangeHandler.h File Reference</td>
<td>1027</td>
<td></td>
</tr>
<tr>
<td>7.57 MamdaOptionContract.h File Reference</td>
<td>1028</td>
<td></td>
</tr>
<tr>
<td>7.58 MamdaOptionContractSet.h File Reference</td>
<td>1029</td>
<td></td>
</tr>
<tr>
<td>7.59 MamdaOptionExchangeUtils.h File Reference</td>
<td>1030</td>
<td></td>
</tr>
<tr>
<td>7.60 MamdaOptionExpirationDateSet.h File Reference</td>
<td>1031</td>
<td></td>
</tr>
<tr>
<td>7.61 MamdaOptionExpiration Strikes.h File Reference</td>
<td>1032</td>
<td></td>
</tr>
<tr>
<td>7.62 MamdaOptionFields.h File Reference</td>
<td>1033</td>
<td></td>
</tr>
<tr>
<td>7.63 MamdaOptionSeriesUpdate.h File Reference</td>
<td>1034</td>
<td></td>
</tr>
<tr>
<td>7.64 MamdaOptionStrikeSet.h File Reference</td>
<td>1035</td>
<td></td>
</tr>
<tr>
<td>7.65 MamdaOptionTypes.h File Reference</td>
<td>1036</td>
<td></td>
</tr>
<tr>
<td>7.66 MamdaOrderBook.h File Reference</td>
<td>1037</td>
<td></td>
</tr>
<tr>
<td>7.67 MamdaOrderBookBasicDelta.h File Reference</td>
<td>1038</td>
<td></td>
</tr>
<tr>
<td>7.68 MamdaOrderBookBasicDeltaList.h File Reference</td>
<td>1039</td>
<td></td>
</tr>
<tr>
<td>7.69 MamdaOrderBookChecker.h File Reference</td>
<td>1040</td>
<td></td>
</tr>
<tr>
<td>7.70 MamdaOrderBookCheckerHandler.h File Reference</td>
<td>1041</td>
<td></td>
</tr>
<tr>
<td>7.71 MamdaOrderBookCheckType.h File Reference</td>
<td>1042</td>
<td></td>
</tr>
<tr>
<td>7.72 MamdaOrderBookClear.h File Reference</td>
<td>1043</td>
<td></td>
</tr>
<tr>
<td>7.73 MamdaOrderBookComplexDelta.h File Reference</td>
<td>1044</td>
<td></td>
</tr>
<tr>
<td>7.74 MamdaOrderBookConcreteComplexDelta.h File Reference</td>
<td>1045</td>
<td></td>
</tr>
<tr>
<td>7.75 MamdaOrderBookConcreteSimpleDelta.h File Reference</td>
<td>1046</td>
<td></td>
</tr>
<tr>
<td>7.76 MamdaOrderBookDelta.h File Reference</td>
<td>1047</td>
<td></td>
</tr>
<tr>
<td>7.77 MamdaOrderBookDepthFilter.h File Reference</td>
<td>1048</td>
<td></td>
</tr>
<tr>
<td>Section</td>
<td>File Name</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>7.78</td>
<td>MamdaOrderBookEntry.h File Reference</td>
<td>1049</td>
</tr>
<tr>
<td>7.79</td>
<td>MamdaOrderBookEntryFilter.h File Reference</td>
<td>1050</td>
</tr>
<tr>
<td>7.80</td>
<td>MamdaOrderBookEntryManager.h File Reference</td>
<td>1051</td>
</tr>
<tr>
<td>7.81</td>
<td>MamdaOrderBookExceptions.h File Reference</td>
<td>1052</td>
</tr>
<tr>
<td>7.82</td>
<td>MamdaOrderBookFields.h File Reference</td>
<td>1053</td>
</tr>
<tr>
<td>7.83</td>
<td>MamdaOrderBookGap.h File Reference</td>
<td>1054</td>
</tr>
<tr>
<td>7.84</td>
<td>MamdaOrderBookHandler.h File Reference</td>
<td>1055</td>
</tr>
<tr>
<td>7.85</td>
<td>MamdaOrderBookListener.h File Reference</td>
<td>1056</td>
</tr>
<tr>
<td>7.86</td>
<td>MamdaOrderBookPriceLevel.h File Reference</td>
<td>1057</td>
</tr>
<tr>
<td>7.87</td>
<td>MamdaOrderBookRecap.h File Reference</td>
<td>1058</td>
</tr>
<tr>
<td>7.88</td>
<td>MamdaOrderBookSimpleDelta.h File Reference</td>
<td>1059</td>
</tr>
<tr>
<td>7.89</td>
<td>MamdaOrderBookTypes.h File Reference</td>
<td>1060</td>
</tr>
<tr>
<td>7.90</td>
<td>MamdaOrderImbalanceFields.h File Reference</td>
<td>1061</td>
</tr>
<tr>
<td>7.91</td>
<td>MamdaOrderImbalanceHandler.h File Reference</td>
<td>1062</td>
</tr>
<tr>
<td>7.92</td>
<td>MamdaOrderImbalanceListener.h File Reference</td>
<td>1063</td>
</tr>
<tr>
<td>7.93</td>
<td>MamdaOrderImbalanceRecap.h File Reference</td>
<td>1064</td>
</tr>
<tr>
<td>7.94</td>
<td>MamdaOrderImbalanceSide.h File Reference</td>
<td>1065</td>
</tr>
<tr>
<td>7.95</td>
<td>MamdaOrderImbalanceType.h File Reference</td>
<td>1066</td>
</tr>
<tr>
<td>7.96</td>
<td>MamdaOrderImbalanceUpdate.h File Reference</td>
<td>1068</td>
</tr>
<tr>
<td>7.97</td>
<td>MamdaPubStatus.h File Reference</td>
<td>1069</td>
</tr>
<tr>
<td>7.98</td>
<td>MamdaPubStatusFields.h File Reference</td>
<td>1070</td>
</tr>
<tr>
<td>7.99</td>
<td>MamdaPubStatusHandler.h File Reference</td>
<td>1071</td>
</tr>
<tr>
<td>7.100</td>
<td>MamdaPubStatusListener.h File Reference</td>
<td>1072</td>
</tr>
<tr>
<td>7.101</td>
<td>MamdaQualityListener.h File Reference</td>
<td>1073</td>
</tr>
<tr>
<td>7.102</td>
<td>MamdaQuery.h File Reference</td>
<td>1074</td>
</tr>
<tr>
<td>7.103</td>
<td>MamdaQuoteChecker.h File Reference</td>
<td>1075</td>
</tr>
<tr>
<td>7.104</td>
<td>MamdaQuoteClosing.h File Reference</td>
<td>1076</td>
</tr>
<tr>
<td>7.105</td>
<td>MamdaQuoteFields.h File Reference</td>
<td>1077</td>
</tr>
<tr>
<td>7.106</td>
<td>MamdaQuoteGap.h File Reference</td>
<td>1078</td>
</tr>
<tr>
<td>7.107</td>
<td>MamdaQuoteHandler.h File Reference</td>
<td>1079</td>
</tr>
<tr>
<td>7.108</td>
<td>MamdaQuoteListener.h File Reference</td>
<td>1080</td>
</tr>
<tr>
<td>7.140</td>
<td>MamdaUncrossPriceInd.h File Reference</td>
<td>1115</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>7.141</td>
<td>MamdaVersion.h File Reference</td>
<td>1116</td>
</tr>
</tbody>
</table>
Chapter 1

MAMDA C++ API Namespace Index

1.1 MAMDA C++ API Namespace List

Here is a list of all namespaces with brief descriptions:

Wombat (MamdaOrderImbalanceUpdate is an interface that provides access to order imbalance related fields) ... ... ... ... ... ... ... ... ... ... ... 23
## Chapter 2

### MAMDA C++ API Hierarchical Index

#### 2.1 MAMDA C++ API Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wombat::char_str_less_than</td>
<td>59</td>
</tr>
<tr>
<td>Wombat::MamdaAuctionFields</td>
<td>61</td>
</tr>
<tr>
<td>Wombat::MamdaAuctionHandler</td>
<td>63</td>
</tr>
<tr>
<td>Wombat::MamdaBasicErrorListener</td>
<td>85</td>
</tr>
<tr>
<td>Wombat::MamdaBasicEvent</td>
<td>86</td>
</tr>
<tr>
<td>Wombat::MamdaAuctionUpdate</td>
<td>81</td>
</tr>
<tr>
<td>Wombat::MamdaAuctionListener</td>
<td>66</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicGap</td>
<td>113</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicListener</td>
<td>130</td>
</tr>
<tr>
<td>Wombat::MamdaConcreteBasicEvent</td>
<td>153</td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyUpdate</td>
<td>183</td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyListener</td>
<td>171</td>
</tr>
<tr>
<td>Wombat::MamdaNewsMetaData</td>
<td>262</td>
</tr>
<tr>
<td>Wombat::MamdaNewsHeadline</td>
<td>238</td>
</tr>
<tr>
<td>Wombat::MamdaOptionSeriesUpdate</td>
<td>332</td>
</tr>
<tr>
<td>Wombat::MamdaOptionChainListener</td>
<td>292</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookClear</td>
<td>403</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookComplexDelta</td>
<td>405</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookConcreteComplexDelta</td>
<td>406</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookDelta</td>
<td>424</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookGap</td>
<td>461</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookRecap</td>
<td>510</td>
</tr>
<tr>
<td>Class/Module</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookSimpleDelta</td>
<td>512</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookConcreteSimpleDelta</td>
<td>415</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceUpdate</td>
<td>565</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceListener</td>
<td>526</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteClosing</td>
<td>602</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteGap</td>
<td>610</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteListener</td>
<td>617</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteOutOfSequence</td>
<td>650</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteListener</td>
<td>617</td>
</tr>
<tr>
<td>Wombat::MamdaQuotePossiblyDuplicate</td>
<td>659</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteListener</td>
<td>617</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteUpdate</td>
<td>704</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteListener</td>
<td>617</td>
</tr>
<tr>
<td>Wombat::MamdaSecStatus</td>
<td>722</td>
</tr>
<tr>
<td>Wombat::MamdaSecStatusListener</td>
<td>739</td>
</tr>
<tr>
<td>Wombat::MamdaTradeCancelOrError</td>
<td>795</td>
</tr>
<tr>
<td>Wombat::MamdaTradeListener</td>
<td>842</td>
</tr>
<tr>
<td>Wombat::MamdaTradeClosing</td>
<td>810</td>
</tr>
<tr>
<td>Wombat::MamdaTradeCorrection</td>
<td>812</td>
</tr>
<tr>
<td>Wombat::MamdaTradeListener</td>
<td>842</td>
</tr>
<tr>
<td>Wombat::MamdaTradeGap</td>
<td>834</td>
</tr>
<tr>
<td>Wombat::MamdaTradeListener</td>
<td>842</td>
</tr>
<tr>
<td>Wombat::MamdaTradeOutOfSequence</td>
<td>906</td>
</tr>
<tr>
<td>Wombat::MamdaTradeListener</td>
<td>842</td>
</tr>
<tr>
<td>Wombat::MamdaTradePossiblyDuplicate</td>
<td>915</td>
</tr>
<tr>
<td>Wombat::MamdaTradeListener</td>
<td>842</td>
</tr>
<tr>
<td>Wombat::MamdaTradeReport</td>
<td>957</td>
</tr>
<tr>
<td>Wombat::MamdaTradeListener</td>
<td>842</td>
</tr>
<tr>
<td>Wombat::MamdaBasicMsgListener</td>
<td>97</td>
</tr>
<tr>
<td>Wombat::MamdaBasicQualityListener</td>
<td>98</td>
</tr>
<tr>
<td>Wombat::MamdaBasicRecap</td>
<td>99</td>
</tr>
<tr>
<td>Wombat::MamdaAuctionRecap</td>
<td>77</td>
</tr>
<tr>
<td>Wombat::MamdaAuctionListener</td>
<td>66</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicLevel</td>
<td>115</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicListener</td>
<td>130</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicLevelEntry</td>
<td>121</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicListener</td>
<td>130</td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyRecap</td>
<td>180</td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyListener</td>
<td>171</td>
</tr>
<tr>
<td>Wombat::MamdaFundamentals</td>
<td>212</td>
</tr>
<tr>
<td>Wombat::MamdaFundamentalListener</td>
<td>199</td>
</tr>
<tr>
<td>Class</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceRecap</td>
<td>545</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceListener</td>
<td>526</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteRecap</td>
<td>668</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteListener</td>
<td>617</td>
</tr>
<tr>
<td>Wombat::MamdaSecStatusRecap</td>
<td>764</td>
</tr>
<tr>
<td>Wombat::MamdaSecStatusListener</td>
<td>739</td>
</tr>
<tr>
<td>Wombat::MamdaTradeRecap</td>
<td>924</td>
</tr>
<tr>
<td>Wombat::MamdaTradeListener</td>
<td>842</td>
</tr>
<tr>
<td>Wombat::MamdaBasicSubscription</td>
<td>106</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicBookHandler</td>
<td>110</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicLevelEntryHandler</td>
<td>126</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicLevelHandler</td>
<td>128</td>
</tr>
<tr>
<td>Wombat::MamdaCheckerHandler</td>
<td>146</td>
</tr>
<tr>
<td>Wombat::MamdaCommonFields</td>
<td>148</td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyFields</td>
<td>166</td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyHandler</td>
<td>168</td>
</tr>
<tr>
<td>Wombat::MamdaDataException</td>
<td>186</td>
</tr>
<tr>
<td>Wombat::MamdaErrorListener</td>
<td>190</td>
</tr>
<tr>
<td>Wombat::MamdaFields</td>
<td>191</td>
</tr>
<tr>
<td>Wombat::MamdaOptionFields</td>
<td>330</td>
</tr>
<tr>
<td>Wombat::MamdaFundamentalFields</td>
<td>192</td>
</tr>
<tr>
<td>Wombat::MamdaFundamentalHandler</td>
<td>197</td>
</tr>
<tr>
<td>Wombat::MamdaLock</td>
<td>220</td>
</tr>
<tr>
<td>Wombat::MamdaMsgListener</td>
<td>222</td>
</tr>
<tr>
<td>Wombat::MamdaAuctionListener</td>
<td>66</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicListener</td>
<td>130</td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyListener</td>
<td>171</td>
</tr>
<tr>
<td>Wombat::MamdaFundamentalListener</td>
<td>199</td>
</tr>
<tr>
<td>Wombat::MamdaMultiParticipantManager</td>
<td>226</td>
</tr>
<tr>
<td>Wombat::MamdaMultiSecurityManager</td>
<td>231</td>
</tr>
<tr>
<td>Wombat::MamdaOptionChainListener</td>
<td>292</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookListener</td>
<td>470</td>
</tr>
<tr>
<td>MamdaOrderBookDepthFilter</td>
<td>426</td>
</tr>
<tr>
<td>MamdaQuoteToBookListener</td>
<td>692</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceListener</td>
<td>526</td>
</tr>
<tr>
<td>Wombat::MamdaPubStatusListener</td>
<td>590</td>
</tr>
<tr>
<td>Wombat::MamdaQuoteListener</td>
<td>617</td>
</tr>
<tr>
<td>Wombat::MamdaSecStatusListener</td>
<td>739</td>
</tr>
<tr>
<td>Wombat::MamdaTradeListener</td>
<td>842</td>
</tr>
<tr>
<td>Wombat::MamdaMultiParticipantHandler</td>
<td>224</td>
</tr>
<tr>
<td>Wombat::MamdaMultiSecurityHandler</td>
<td>229</td>
</tr>
<tr>
<td>Wombat::MamdaNewsFields</td>
<td>234</td>
</tr>
<tr>
<td>Wombat::MamdaNewsHeadlineHandler</td>
<td>255</td>
</tr>
<tr>
<td>Wombat::MamdaNewsManager</td>
<td>257</td>
</tr>
<tr>
<td>Class</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Wombat::MamdaNewsQueryHandler</td>
<td>268</td>
</tr>
<tr>
<td>Wombat::MamdaNewsStory</td>
<td>270</td>
</tr>
<tr>
<td>Wombat::MamdaNewsStoryHandler</td>
<td>276</td>
</tr>
<tr>
<td>Wombat::MamdaOptionChain</td>
<td>278</td>
</tr>
<tr>
<td>Wombat::MamdaOptionChain::const_iterator</td>
<td>284</td>
</tr>
<tr>
<td>Wombat::MamdaOptionChain::iterator</td>
<td>286</td>
</tr>
<tr>
<td>Wombat::MamdaOptionChainHandler</td>
<td>288</td>
</tr>
<tr>
<td>Wombat::MamdaOptionChainView</td>
<td>302</td>
</tr>
<tr>
<td>Wombat::MamdaOptionChainViewRangeHandler</td>
<td>309</td>
</tr>
<tr>
<td>Wombat::MamdaOptionContract</td>
<td>310</td>
</tr>
<tr>
<td>Wombat::MamdaOptionExchangeUtils</td>
<td>324</td>
</tr>
<tr>
<td>Wombat::MamdaOptionStrikeSet</td>
<td>334</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBook</td>
<td>337</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBook::askEntryIterator</td>
<td>365</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBook::askIterator</td>
<td>369</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBook::bidEntryIterator</td>
<td>373</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBook::bidIterator</td>
<td>377</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookBasicDelta</td>
<td>381</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookSimpleDelta</td>
<td>512</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookBasicDeltaList</td>
<td>388</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookComplexDelta</td>
<td>405</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookBasicDeltaList::iterator</td>
<td>394</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookChecker</td>
<td>398</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookCheckerHandler</td>
<td>401</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookEntry</td>
<td>434</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookEntryFilter</td>
<td>448</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookEntryManager</td>
<td>449</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookException</td>
<td>452</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookDuplicateEntry</td>
<td>432</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookInvalidEntry</td>
<td>468</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookMissingEntry</td>
<td>485</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookFields</td>
<td>454</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookHandler</td>
<td>463</td>
</tr>
<tr>
<td>MamdaOrderBookDepthFilter</td>
<td>426</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookPriceLevel</td>
<td>487</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookPriceLevel::iterator</td>
<td>506</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookTypes</td>
<td>514</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceFields</td>
<td>518</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceHandler</td>
<td>523</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImalanceSide</td>
<td>559</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceType</td>
<td>561</td>
</tr>
<tr>
<td>Wombat::MamdaPubStatus</td>
<td>580</td>
</tr>
<tr>
<td>Wombat::MamdaPubStatusListener</td>
<td>590</td>
</tr>
<tr>
<td>Wombat::MamdaPubStatusFields</td>
<td>584</td>
</tr>
</tbody>
</table>
2.1 MAMDA C++ API Class Hierarchy

Wombat::MamdaPubStatusHandler ........................................ 588
Wombat::MamdaQualityListener .............................................. 596
Wombat::MamdaQuery .......................................................... 597
  Wombat::MamdaAndQuery ................................................... 60
  Wombat::MamdaContainsAllQuery ......................................... 162
  Wombat::MamdaContainsQuery ........................................... 164
  Wombat::MamdaDateQuery .................................................. 187
  Wombat::MamdaEqualsQuery ............................................... 188
  Wombat::MamdaOrQuery ..................................................... 579
Wombat::MamdaQuoteChecker ................................................ 599
Wombat::MamdaQuoteFields ................................................... 605
Wombat::MamdaQuoteHandler ................................................ 613
Wombat::MamdaSecStatusFields ............................................ 732
Wombat::MamdaSecStatusHandler .......................................... 737
  Wombat::MamdaSecStatusSymbolSourceAdapter .......................... 779
Wombat::MamdaSubscription ................................................ 782
Wombat::MamdaSymbolSourceEvent ........................................ 792
  Wombat::MamdaSymbolSourceAdapter .................................... 779
Wombat::MamdaSymbolSourceHandler ...................................... 794
Wombat::MamdaTradeChecker ................................................ 807
Wombat::MamdaTradeFields .................................................. 828
Wombat::MamdaTradeHandler ................................................ 837
map .............................................................................. 969
  Wombat::MamdaOptionContractSet ....................................... 322
  Wombat::MamdaOptionExpirationDateSet ................................ 326
  Wombat::MamdaOptionExpirationStrikes ................................ 328
Chapter 3

MAMDA C++ API Class Index

3.1 MAMDA C++ API Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

- `Wombat::char_str_less_than` .................................................. 59
- `Wombat::MamdaAndQuery` .................................................... 60
- `Wombat::MamdaAuctionFields` .............................................. 61
- `Wombat::MamdaAuctionHandler` (MamdaAuctionHandler is an interface for applications that want to have an easy way to access currency data) ....................... 63
- `Wombat::MamdaAuctionListener` (MamdaAuctionListener is a class that specializes in handling currency data. Developers provide their own implementation of the MamdaAuctionHandler interface and will be delivered notifications for updates in the currency data) ......................... 66
- `Wombat::MamdaAuctionRecap` (MamdaAuctionRecap is an interface that provides access to the currency related fields) .......................... 77
- `Wombat::MamdaAuctionUpdate` (MamdaAuctionUpdate is an interface that provides access to the currency related fields) .......................... 81
- `Wombat::MamdaBasicErrorListener` (MamdaBasicErrorListener defines an interface for handling error notifications for a MamdaBasicSubscription) ......................... 85
- `Wombat::MamdaBasicEvent` (MamdaBasicEvent is a superclass interface that provides access to common event related fields) ......................... 86
- `Wombat::MamdaBasicMsgListener` (MamdaBasicMsgListener defines an interface for handling MAMA messages for a MamdaBasicSubscription) ......................... 97
- `Wombat::MamdaBasicQualityListener` (MamdaBasicQualityListener defines an interface for handling changes in quality notifications for a MamdaBasicSubscription) ......................... 98
<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wombat::MamdaBasicRecap</td>
<td>MamdaBasicRecap is an interface that provides access to recap related fields</td>
</tr>
<tr>
<td>Wombat::MamdaBasicSubscription</td>
<td>A MamdaBasicSubscription is used to register interest in a particular symbol</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicBookHandler</td>
<td>MamdaBookAtomicBookHandler is an interface for applications that need to know when a MamdaBookAtomicListener finishes processing a single book update</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicGap</td>
<td>MamdaBookAtomicGap is an interface that provides access to order book atomic update gap related fields</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicLevel</td>
<td>MamdaBookAtomicLevel is an interface that provides access to trade related fields</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicLevelEntry</td>
<td>MamdaBookAtomicLevel is an interface that provides access to Price Level and Price Level Entry fields</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicLevelEntryHandler</td>
<td>MamdaBookAtomicLevelEntryHandler is an interface for applications that want to have an easy way to handle order book Price Level &amp; Entry updates</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicLevelHandler</td>
<td>MamdaBookAtomicLevelHandler is an interface for applications that want to have an easy way to handle order book Price Level updates</td>
</tr>
<tr>
<td>Wombat::MamdaBookAtomicListener</td>
<td>MamdaBookAtomicListener is a class that specializes in handling order book updates</td>
</tr>
<tr>
<td>Wombat::MamdaCheckerHandler</td>
<td>MamdaCheckerHandler is an interface for applications that want to handle the results of the MamdaQuoteChecker and MamdaTradeChecker</td>
</tr>
<tr>
<td>Wombat::MamdaCommonFields</td>
<td>Utility cache of MamaFieldDescriptors which are used internally by the API when accessing common fields from update messages</td>
</tr>
<tr>
<td>Wombat::MamdaConcreteBasicEvent</td>
<td>MamdaConcreteBasicEvent is intended to be used to help implement concrete versions of various classes derived from MamdaBasicEvent</td>
</tr>
<tr>
<td>Wombat::MamdaContainsAllQuery</td>
<td></td>
</tr>
<tr>
<td>Wombat::MamdaContainsQuery</td>
<td></td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyFields</td>
<td></td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyHandler</td>
<td>MamdaCurrencyHandler is an interface for applications that want to have an easy way to access currency data</td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyListener</td>
<td>MamdaCurrencyListener is a class that specializes in handling currency data, Developers provide their own implementation of the MamdaCurrencyHandler interface and will be delivered notifications for updates in the currency data</td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyRecap</td>
<td>MamdaCurrencyRecap is an interface that provides access to the currency related fields</td>
</tr>
<tr>
<td>Wombat::MamdaCurrencyUpdate</td>
<td>MamdaCurrencyUpdate is an interface that provides access to the currency related fields</td>
</tr>
<tr>
<td>Class/Definition</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Wombat::MamdaDataException (MAMDA data exceptions)</td>
<td>186</td>
</tr>
<tr>
<td>Wombat::MamdaDateQuery</td>
<td>187</td>
</tr>
<tr>
<td>Wombat::MamdaEqualsQuery</td>
<td>188</td>
</tr>
<tr>
<td>Wombat::MamdaErrorListener (MamdaErrorListener defines an interface for handling error notifications for a MamdaSubscription)</td>
<td>190</td>
</tr>
<tr>
<td>Wombat::MamdaFields</td>
<td>191</td>
</tr>
<tr>
<td>Wombat::MamdaFundamentalFields</td>
<td>192</td>
</tr>
<tr>
<td>Wombat::MamdaFundamentalHandler (MamdaFundamentalHandler is an interface for applications that want to have an easy way to access fundamental equity pricing/analysis attributes, indicators and ratios)</td>
<td>197</td>
</tr>
<tr>
<td>Wombat::MamdaFundamentalListener (MamdaFundamentalListener is a class that specializes in handling fundamental equity pricing/analysis attributes, indicators and ratios)</td>
<td>199</td>
</tr>
<tr>
<td>Wombat::MamdaFundamentals (MamdaFundamentals is an interface that provides access to the fundamental equity pricing/analysis attributes, indicators and ratios)</td>
<td>212</td>
</tr>
<tr>
<td>Wombat::MamdaLock</td>
<td>220</td>
</tr>
<tr>
<td>Wombat::MamdaMsgListener (MamdaMsgListener defines an interface for handling MAMA messages for a MamdaSubscription)</td>
<td>222</td>
</tr>
<tr>
<td>Wombat::MamdaMultiParticipantHandler (The MamdaMultiParticipantHandler class is an interface that allows a developer to be notified dynamically when participants are added to the list)</td>
<td>224</td>
</tr>
<tr>
<td>Wombat::MamdaMultiParticipantManager (MamdaMultiParticipantManager is a class that manages updates on a consolidated basis for securities that may be traded on multiple exchanges and which may have a national best bid and offer)</td>
<td>226</td>
</tr>
<tr>
<td>Wombat::MamdaMultiSecurityHandler (The MamdaMultiSecurityHandler class is an interface that allows a developer to be notified dynamically when securities are added to the list)</td>
<td>229</td>
</tr>
<tr>
<td>Wombat::MamdaMultiSecurityManager (MamdaMultiSecurityManager is a class that manages updates on an arbitrary number of securities that may be traded on multiple exchanges)</td>
<td>231</td>
</tr>
<tr>
<td>Wombat::MamdaNewsFields</td>
<td>234</td>
</tr>
<tr>
<td>Wombat::MamdaNewsHeadline (MamdaNewsHeadline represents a news headline and includes information about many types of meta-data attributes associated with the headline)</td>
<td>238</td>
</tr>
<tr>
<td>Wombat::MamdaNewsHeadlineHandler (MamdaNewsHeadlineHandler is an interface for applications that want to have an easy way to handle news headlines)</td>
<td>255</td>
</tr>
<tr>
<td>Wombat::MamdaNewsManager (MamdaNewsManager provides a class for managing access to streaming news headlines, headline queries, individual story queries, etc)</td>
<td>257</td>
</tr>
<tr>
<td>Wombat::MamdaNewsMetaData (MamdaNewsMetaData represents information about many types of attributes associated with the news headline)</td>
<td>262</td>
</tr>
</tbody>
</table>
Wombat::MamdaNewsQueryHandler (MamdaNewsHandler is an interface for applications that want to have an easy way to handle news queries) ................................................................. 268
Wombat::MamdaNewsStory (MamdaNewsStory represents a complete text of a news story) ................................................................. 270
Wombat::MamdaNewsStoryHandler (MamdaNewsHandler is an interface for applications that want to have an easy way to handle news stories) ................................................................. 276
Wombat::MamdaOptionChain (MamdaOptionChain is a specialized class to represent market data option chains) ................................................................. 278
Wombat::MamdaOptionChain::const_iterator ................................................................. 284
Wombat::MamdaOptionChain::iterator ................................................................. 286
Wombat::MamdaOptionChainHandler (Subclasses of this interface can be registered with the MamdaOptionChainListener in order to receive callbacks whenever the state of the underlying option chain changes on receipt of options updates) ................................................................. 288
Wombat::MamdaOptionChainListener (MamdaOptionChainListener is a class that specializes in handling and managing option chain updates) ................................................................. 292
Wombat::MamdaOptionChainView (A class that represents a "view" of a subset of an option chain) ................................................................. 302
Wombat::MamdaOptionChainViewRangeHandler (Class to handle change in a MamdaOptionChainView range) ................................................................. 309
Wombat::MamdaOptionContract (A class that represents a single option contract) ................................................................. 310
Wombat::MamdaOptionContractSet (A class that represents a set of option contracts at a given strike price) ................................................................. 322
Wombat::MamdaOptionExchangeUtils (A class with static utility functions for dealing with exchanges) ................................................................. 324
Wombat::MamdaOptionExpirationDateSet (A class that represents a set of expiration dates, each of which contains a set of strike prices, each of which contains a set of option contracts, each of which contains exchange-specific contracts) ................................................................. 326
Wombat::MamdaOptionExpirationStrikes (A class that represents a set of strike prices at a particular expiration date) ................................................................. 328
Wombat::MamdaOptionFields (Utility cache of MamaFieldDescriptors which are used internally by the API when accessing options related fields from update messages) ................................................................. 330
Wombat::MamdaOptionSeriesUpdate (MamdaOptionSeriesUpdate is an interface that provides access to fields related to option series update events) ................................................................. 332
Wombat::MamdaOptionStrikeSet (A class that represents the call and put contract sets at a given strike price) ................................................................. 334
Wombat::MamdaOrderBook (MamdaOrderBook is a class that provides order book functionality, including iterators over price levels and entries within price levels) ................................................................. 337
<table>
<thead>
<tr>
<th>Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wombat::MamdaOrderBookFields</td>
<td>Utility cache of MamaFieldDescriptors which are used internally by the API when accessing orderbook related fields from update messages</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookGap</td>
<td>MamdaOrderBookGap is an interface that provides access to order book gap related fields</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookHandler</td>
<td>MamdaOrderBookHandler is an interface for applications that want to have an easy way to handle order book updates</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookInvalidEntry</td>
<td>MamdaOrderBookInvalidEntry is generated when an entry is applied as an update to or deletion from an order book but the internal references to price level and/or order book do not exist</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookListener</td>
<td>MamdaOrderBookListener is a class that specializes in handling order book updates</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookMissingEntry</td>
<td>MamdaOrderBookMissingEntry is generated when an expected entry is not found when updating a MamdaOrderBook or MamdaOrderBookEntryManager</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookPriceLevel</td>
<td>MamdaOrderBookPriceLevel is a class that provides a price level type for order books</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookPriceLevel::iterator</td>
<td></td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookRecap</td>
<td>MamdaOrderBookRecap is an interface that provides access to order book related fields</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookSimpleDelta</td>
<td>MamdaOrderBookSimpleDelta is a class that saves information about a simple order book delta</td>
</tr>
<tr>
<td>Wombat::MamdaOrderBookTypes</td>
<td>MamdaOrderBookTypes is a class that provides order book related typed</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceFields</td>
<td>Utility cache of MamaFieldDescriptors which are used internally by the API when accessing imbalance related fields from update messages</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceHandler</td>
<td>MamdaOrderImbalanceHandler is an interface for applications that want to have an easy way to handle order imbalance updates</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceListener</td>
<td>MamdaOrderImbalanceListener is a class that specializes in handling an imbalance order updates</td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceRecap</td>
<td></td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceSide</td>
<td></td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceType</td>
<td></td>
</tr>
<tr>
<td>Wombat::MamdaOrderImbalanceUpdate</td>
<td></td>
</tr>
<tr>
<td>Wombat::MamdaOrQuery</td>
<td></td>
</tr>
<tr>
<td>Wombat::MamdaPubStatus</td>
<td>MamdaPubStatus is an interface that provides access to the Security Status fields such as symbol announce messages</td>
</tr>
<tr>
<td>Wombat::MamdaPubStatusFields</td>
<td></td>
</tr>
</tbody>
</table>

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
Wombat::MamdaPubStatusHandler (MamdaPubStatusHandler is an interface for applications that want to have an easy way to handle feed handler publisher status updates) ........................................ 588
Wombat::MamdaPubStatusListener (MamdaPubStatusListener is a class that specializes in handling Publisher (Feed Handler) Status updates) ........ 590
Wombat::MamdaQualityListener (MamdaQualityListener defines an interface for handling changes in quality notifications for a MamdaSubscription) ................................................................. 596
Wombat::MamdaQuery ................................................................. 597
Wombat::MamdaQuoteChecker (MamdaQuoteChecker is a class that provides quotes sanity checking by periodically requesting snapshots of the quotes from the publisher and comparing that with an quotes being maintained in real time) ............................................. 599
Wombat::MamdaQuoteClosing (MamdaQuoteClosing is an interface that provides access to quote closing related fields) ............................ 602
Wombat::MamdaQuoteFields (Utility cache of MamaFieldDescriptors which are used internally by the API when accessing quote related fields from update messages) .......................... 605
Wombat::MamdaQuoteGap (MamdaQuoteGap is an interface that provides access to quote gap related fields) ........................................... 610
Wombat::MamdaQuoteHandler (MamdaQuoteHandler is an interface for applications that want to have an easy way to handle quote updates) 613
Wombat::MamdaQuoteListener (MamdaQuoteListener is a class that specializes in handling quote updates) ........................................... 617
Wombat::MamdaQuoteOutOfSequence (MamdaQuoteOutOfSequence is an interface that provides access to fields related to quote updates) 650
Wombat::MamdaQuotePossiblyDuplicate (MamdaQuotePossiblyDuplicate is an interface that provides access to fields related to quote updates which are possible duplicates of previous quote updates) .... 659
Wombat::MamdaQuoteRecap (MamdaQuoteRecap is an interface that provides access to quote related fields) ........................................... 668
MamdaQuoteToBookListener ....................................................... 692
Wombat::MamdaQuoteUpdate (MamdaQuoteUpdate is an interface that provides access to fields related to quote updates) ..................... 704
Wombat::MamdaSecStatus (MamdaSecStatus is an interface that provides access to the Security Status fields such as symbol announce messages) .................................................... 722
Wombat::MamdaSecStatusFields (Utility cache of MamaFieldDescriptors which are used internally by the API when accessing security status related fields from update messages) ........... 732
Wombat::MamdaSecStatusHandler (MamdaSecStatusHandler is an interface for applications that want to have an easy way to handle security status updates) .................................................. 737
Wombat::MamdaSecStatusListener (MamdaSecurityStatusListener is a class that specializes in handling security status updates) ........ 739
Wombat::MamdaSecStatusRecap (MamdaSecStatus is an interface that provides access to the Security Status fields such as symbol announce messages) ................................................. 764
Wombat::MamdaSecStatusSymbolSourceAdapter (MamdaSecStatusSymbolSourceAdapter is a simple adapter class that can be added as a handler to the MamdaSecStatusListener turning it into a MamdaSymbolSource) .................................................. 779
Wombat::MamdaSubscription (A MamdaSubscription is used to register interest in a particular symbol and source) ......................... 782
Wombat::MamdaSymbolSourceEvent (MamdaSymbolSourceEvent is an interface that provides access to a sourced symbol name) ........ 792
Wombat::MamdaSymbolSourceHandler (MamdaSymbolSourceHandler is an interface for applications that want to have an easy way to handle newly sourced symbol events) .......................... 794
Wombat::MamdaTradeCancelOrError (MamdaTradeCancelOrError is an interface that provides access to trade cancellation related fields) 795
Wombat::MamdaTradeChecker (MamdaTradeChecker is a class that provides trades sanity checking by periodically requesting snapshots of the trades from the publisher and comparing that with an trades being maintained in real time) ........................................... 807
Wombat::MamdaTradeClosing (MamdaTradeClosing is an interface that provides access to trade closing related fields) ............. 810
Wombat::MamdaTradeCorrection (MamdaTradeCorrection is an interface that provides access to trade correction related fields) .... 812
Wombat::MamdaTradeFields (Utility cache of MamaFieldDescriptors which are used internally by the API when accessing trade related fields from update messages) ................................. 828
Wombat::MamdaTradeGap (MamdaTradeGap is an interface that provides access to trade gap related fields) .............................. 834
Wombat::MamdaTradeHandler (MamdaTradeHandler is an interface for applications that want to have an easy way to handle trade updates) ................................................................. 837
Wombat::MamdaTradeListener (MamdaTradeListener is a class that specializes in handling trade updates) .................................. 842
Wombat::MamdaTradeOutOfSequence (MamdaTradeOutOfSequence is an interface that provides access to fields related to trade updates which have been identified as being out of sequence with previous update (e.g)) ......................................................... 906
Wombat::MamdaTradePossiblyDuplicate (MamdaTradePossiblyDuplicate is an interface that provides access to fields related to trade updates which have been identified as being possible duplicates of previous updates) ................................. 915
Wombat::MamdaTradeRecap (MamdaTradeRecap is an interface that provides access to trade related fields) ................................. 924
Wombat::MamdaTradeReport (MamdaTradeReport is an interface that provides access to fields related to a trade report)............... 957
map ......................................................... 969
Chapter 4

MAMDA C++ API File Index

4.1 MAMDA C++ API File List

Here is a list of all files with brief descriptions:

- MamdaAuctionFields.h .................................................. 971
- MamdaAuctionHandler.h ............................................... 972
- MamdaAuctionListener.h .............................................. 973
- MamdaAuctionRecap.h .................................................. 974
- MamdaAuctionUpdate.h ................................................ 975
- MamdaBasicEvent.h ..................................................... 976
- MamdaBasicRecap.h ..................................................... 977
- MamdaBasicSubscription.h .......................................... 978
- MamdaBookAtomicBookHandler.h .................................. 979
- MamdaBookAtomicGap.h ............................................... 980
- MamdaBookAtomicLevel.h .......................................... 981
- MamdaBookAtomicLevelEntry.h ................................... 982
- MamdaBookAtomicLevelEntryHandler.h ......................... 983
- MamdaBookAtomicLevelHandler.h ................................. 984
- MamdaBookAtomicListener.h ....................................... 985
- MamdaCheckerHandler.h ............................................. 986
- MamdaCheckerType.h ................................................ 987
- MamdaCommonFields.h ............................................... 988
- MamdaConcreteBasicEvent.h ...................................... 989
- MamdaConfig.h ......................................................... 990
- MamdaCurrencyFields.h ............................................. 991
- MamdaCurrencyHandler.h .......................................... 992
- MamdaCurrencyListener.h ........................................... 993
- MamdaCurrencyRecap.h .............................................. 994
- MamdaCurrencyUpdate.h ............................................ 995
<table>
<thead>
<tr>
<th>File Name</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>MamdaDataException.h</td>
<td>996</td>
</tr>
<tr>
<td>MamdaErrorListener.h</td>
<td>997</td>
</tr>
<tr>
<td>MamdaFields.h</td>
<td>998</td>
</tr>
<tr>
<td>MamdaFieldState.h</td>
<td>999</td>
</tr>
<tr>
<td>MamdaFundamentalFields.h</td>
<td>1000</td>
</tr>
<tr>
<td>MamdaFundamentalHandler.h</td>
<td>1001</td>
</tr>
<tr>
<td>MamdaFundamentalListener.h</td>
<td>1002</td>
</tr>
<tr>
<td>MamdaFundamentals.h</td>
<td>1003</td>
</tr>
<tr>
<td>MamdaLock.h</td>
<td>1004</td>
</tr>
<tr>
<td>MamdaMsgListener.h</td>
<td>1006</td>
</tr>
<tr>
<td>MamdaMultiParticipantHandler.h</td>
<td>1007</td>
</tr>
<tr>
<td>MamdaMultiParticipantManager.h</td>
<td>1008</td>
</tr>
<tr>
<td>MamdaMultiSecurityHandler.h</td>
<td>1009</td>
</tr>
<tr>
<td>MamdaMultiSecurityManager.h</td>
<td>1010</td>
</tr>
<tr>
<td>MamdaNewsFields.h</td>
<td>1011</td>
</tr>
<tr>
<td>MamdaNewsHeadline.h</td>
<td>1012</td>
</tr>
<tr>
<td>MamdaNewsHeadlineHandler.h</td>
<td>1013</td>
</tr>
<tr>
<td>MamdaNewsManager.h</td>
<td>1014</td>
</tr>
<tr>
<td>MamdaNewsMetaData.h</td>
<td>1015</td>
</tr>
<tr>
<td>MamdaNewsQuery.h</td>
<td>1016</td>
</tr>
<tr>
<td>MamdaNewsQueryHandler.h</td>
<td>1017</td>
</tr>
<tr>
<td>MamdaNewsStory.h</td>
<td>1018</td>
</tr>
<tr>
<td>MamdaNewsStoryHandler.h</td>
<td>1019</td>
</tr>
<tr>
<td>MamdaNewsTypes.h</td>
<td>1020</td>
</tr>
<tr>
<td>MamdaOptionalConfig.h</td>
<td>1021</td>
</tr>
<tr>
<td>MamdaOptionAtTheMoneyCompareType.h</td>
<td>1022</td>
</tr>
<tr>
<td>MamdaOptionChain.h</td>
<td>1023</td>
</tr>
<tr>
<td>MamdaOptionChainHandler.h</td>
<td>1024</td>
</tr>
<tr>
<td>MamdaOptionChainListener.h</td>
<td>1025</td>
</tr>
<tr>
<td>MamdaOptionChainView.h</td>
<td>1026</td>
</tr>
<tr>
<td>MamdaOptionChainViewRangeHandler.h</td>
<td>1027</td>
</tr>
<tr>
<td>MamdaOptionContract.h</td>
<td>1028</td>
</tr>
<tr>
<td>MamdaOptionContractSet.h</td>
<td>1029</td>
</tr>
<tr>
<td>MamdaOptionExchangeUtils.h</td>
<td>1030</td>
</tr>
<tr>
<td>MamdaOptionExpirationDateSet.h</td>
<td>1031</td>
</tr>
<tr>
<td>MamdaOptionExpirationStrikes.h</td>
<td>1032</td>
</tr>
<tr>
<td>MamdaOptionFields.h</td>
<td>1033</td>
</tr>
<tr>
<td>MamdaOptionSeriesUpdate.h</td>
<td>1034</td>
</tr>
<tr>
<td>MamdaOptionStrikeSet.h</td>
<td>1035</td>
</tr>
<tr>
<td>MamdaOptionTypes.h</td>
<td>1036</td>
</tr>
<tr>
<td>MamdaOrderBook.h</td>
<td>1037</td>
</tr>
<tr>
<td>MamdaOrderBookBasicDelta.h</td>
<td>1038</td>
</tr>
<tr>
<td>MamdaOrderBookBasicDeltaList.h</td>
<td>1039</td>
</tr>
<tr>
<td>MamdaOrderBookChecker.h</td>
<td>1040</td>
</tr>
<tr>
<td>MamdaOrderBookCheckerHandler.h</td>
<td>1041</td>
</tr>
<tr>
<td>MamdaOrderBookCheckType.h</td>
<td>1042</td>
</tr>
<tr>
<td>File Name</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>MamdaOrderBookClear.h</td>
<td>1043</td>
</tr>
<tr>
<td>MamdaOrderBookComplexDelta.h</td>
<td>1044</td>
</tr>
<tr>
<td>MamdaOrderBookConcreteComplexDelta.h</td>
<td>1045</td>
</tr>
<tr>
<td>MamdaOrderBookConcreteSimpleDelta.h</td>
<td>1046</td>
</tr>
<tr>
<td>MamdaOrderBookDelta.h</td>
<td>1047</td>
</tr>
<tr>
<td>MamdaOrderBookDepthFilter.h</td>
<td>1048</td>
</tr>
<tr>
<td>MamdaOrderBookEntry.h</td>
<td>1049</td>
</tr>
<tr>
<td>MamdaOrderBookEntryFilter.h</td>
<td>1050</td>
</tr>
<tr>
<td>MamdaOrderBookEntryManager.h</td>
<td>1051</td>
</tr>
<tr>
<td>MamdaOrderBookExceptions.h</td>
<td>1052</td>
</tr>
<tr>
<td>MamdaOrderBookFields.h</td>
<td>1053</td>
</tr>
<tr>
<td>MamdaOrderBookGap.h</td>
<td>1054</td>
</tr>
<tr>
<td>MamdaOrderBookHandler.h</td>
<td>1055</td>
</tr>
<tr>
<td>MamdaOrderBookListener.h</td>
<td>1056</td>
</tr>
<tr>
<td>MamdaOrderBookPriceLevel.h</td>
<td>1057</td>
</tr>
<tr>
<td>MamdaOrderBookRecap.h</td>
<td>1058</td>
</tr>
<tr>
<td>MamdaOrderBookSimpleDelta.h</td>
<td>1059</td>
</tr>
<tr>
<td>MamdaOrderBookTypes.h</td>
<td>1060</td>
</tr>
<tr>
<td>MamdaOrderImbalanceFields.h</td>
<td>1061</td>
</tr>
<tr>
<td>MamdaOrderImbalanceHandler.h</td>
<td>1062</td>
</tr>
<tr>
<td>MamdaOrderImbalanceListener.h</td>
<td>1063</td>
</tr>
<tr>
<td>MamdaOrderImbalanceRecap.h</td>
<td>1064</td>
</tr>
<tr>
<td>MamdaOrderImbalanceSide.h</td>
<td>1065</td>
</tr>
<tr>
<td>MamdaOrderImbalanceType.h</td>
<td>1066</td>
</tr>
<tr>
<td>MamdaOrderImbalanceUpdate.h</td>
<td>1068</td>
</tr>
<tr>
<td>MamdaPubStatus.h</td>
<td>1069</td>
</tr>
<tr>
<td>MamdaPubStatusFields.h</td>
<td>1070</td>
</tr>
<tr>
<td>MamdaPubStatusHandler.h</td>
<td>1071</td>
</tr>
<tr>
<td>MamdaPubStatusListener.h</td>
<td>1072</td>
</tr>
<tr>
<td>MamdaQualityListener.h</td>
<td>1073</td>
</tr>
<tr>
<td>MamdaQuery.h</td>
<td>1074</td>
</tr>
<tr>
<td>MamdaQuoteChecker.h</td>
<td>1075</td>
</tr>
<tr>
<td>MamdaQuoteClosing.h</td>
<td>1076</td>
</tr>
<tr>
<td>MamdaQuoteFields.h</td>
<td>1077</td>
</tr>
<tr>
<td>MamdaQuoteGap.h</td>
<td>1078</td>
</tr>
<tr>
<td>MamdaQuoteHandler.h</td>
<td>1079</td>
</tr>
<tr>
<td>MamdaQuoteListener.h</td>
<td>1080</td>
</tr>
<tr>
<td>MamdaQuoteOutOfSequence.h</td>
<td>1081</td>
</tr>
<tr>
<td>MamdaQuotePossiblyDuplicate.h</td>
<td>1082</td>
</tr>
<tr>
<td>MamdaQuoteRecap.h</td>
<td>1083</td>
</tr>
<tr>
<td>MamdaQuoteToBookListener.h</td>
<td>1084</td>
</tr>
<tr>
<td>MamdaQuoteUpdate.h</td>
<td>1085</td>
</tr>
<tr>
<td>MamdaSecStatus.h</td>
<td>1086</td>
</tr>
<tr>
<td>MamdaSecStatusFields.h</td>
<td>1087</td>
</tr>
<tr>
<td>MamdaSecStatusHandler.h</td>
<td>1088</td>
</tr>
<tr>
<td>MamdaSecStatusListener.h</td>
<td>1089</td>
</tr>
<tr>
<td>C++ File Name</td>
<td>Page</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>MamdaSecStatusRecap.h</td>
<td>1090</td>
</tr>
<tr>
<td>MamdaSecStatusSymbolSourceAdapter.h</td>
<td>1091</td>
</tr>
<tr>
<td>MamdaSecurityStatus.h</td>
<td>1092</td>
</tr>
<tr>
<td>MamdaSecurityStatusQual.h</td>
<td>1093</td>
</tr>
<tr>
<td>MamdaSubscription.h</td>
<td>1097</td>
</tr>
<tr>
<td>MamdaSymbolSourceEvent.h</td>
<td>1098</td>
</tr>
<tr>
<td>MamdaSymbolSourceHandler.h</td>
<td>1099</td>
</tr>
<tr>
<td>MamdaTradeCancelOrError.h</td>
<td>1100</td>
</tr>
<tr>
<td>MamdaTradeChecker.h</td>
<td>1101</td>
</tr>
<tr>
<td>MamdaTradeClosing.h</td>
<td>1102</td>
</tr>
<tr>
<td>MamdaTradeCorrection.h</td>
<td>1103</td>
</tr>
<tr>
<td>MamdaTradeDirection.h</td>
<td>1104</td>
</tr>
<tr>
<td>MamdaTradeExecVenue.h</td>
<td>1105</td>
</tr>
<tr>
<td>MamdaTradeFields.h</td>
<td>1106</td>
</tr>
<tr>
<td>MamdaTradeGap.h</td>
<td>1107</td>
</tr>
<tr>
<td>MamdaTradeHandler.h</td>
<td>1108</td>
</tr>
<tr>
<td>MamdaTradeListener.h</td>
<td>1109</td>
</tr>
<tr>
<td>MamdaTradeOutOfSequence.h</td>
<td>1110</td>
</tr>
<tr>
<td>MamdaTradePossiblyDuplicate.h</td>
<td>1111</td>
</tr>
<tr>
<td>MamdaTradeRecap.h</td>
<td>1112</td>
</tr>
<tr>
<td>MamdaTradeReport.h</td>
<td>1113</td>
</tr>
<tr>
<td>MamdaTradeSide.h</td>
<td>1114</td>
</tr>
<tr>
<td>MamdaUncrossPriceInd.h</td>
<td>1115</td>
</tr>
<tr>
<td>MamdaVersion.h</td>
<td>1116</td>
</tr>
</tbody>
</table>
Chapter 5

MAMDA C++ API Namespace Documentation

5.1 Wombat Namespace Reference

*MamdaOrderImbalanceUpdate* is an interface that provides access to order imbalance related fields.

**Classes**

- class *MamdaAuctionFields*
- class *MamdaAuctionHandler*
  
  *MamdaAuctionHandler* is an interface for applications that want to have an easy way to access currency data.

- class *MamdaAuctionListener*
  
  *MamdaAuctionListener* is a class that specializes in handling currency data. Developers provide their own implementation of the *MamdaAuctionHandler* interface and will be delivered notifications for updates in the currency data.

- class *MamdaAuctionRecap*
  
  *MamdaAuctionRecap* is an interface that provides access to the currency related fields.

- class *MamdaAuctionUpdate*
  
  *MamdaAuctionUpdate* is an interface that provides access to the currency related fields.
• class MamdaBasicEvent
  MamdaBasicEvent is a superclass interface that provides access to common event related fields.

• class MamdaBasicRecap
  MamdaBasicRecap is an interface that provides access to recap related fields.

• class MamdaBasicSubscription
  A MamdaBasicSubscription is used to register interest in a particular symbol.

• class MamdaCheckerHandler
  MamdaCheckerHandler is an interface for applications that want to handle the results of the MamdaQuoteChecker and MamdaTradeChecker.

• class MamdaCommonFields
  Utility cache of MamaFieldDescriptors which are used internally by the API when accessing common fields from update messages.

• class MamdaConcreteBasicEvent
  MamdaConcreteBasicEvent is intended to be used to help implement concrete versions of various classes derived from MamdaBasicEvent.

• class MamdaCurrencyFields

• class MamdaCurrencyHandler
  MamdaCurrencyHandler is an interface for applications that want to have an easy way to access currency data.

• class MamdaCurrencyListener
  MamdaCurrencyListener is a class that specializes in handling currency data. Developers provide their own implementation of the MamdaCurrencyHandler interface and will be delivered notifications for updates in the currency data.

• class MamdaCurrencyRecap
  MamdaCurrencyRecap is an interface that provides access to the currency related fields.

• class MamdaCurrencyUpdate
  MamdaCurrencyUpdate is an interface that provides access to the currency related fields.

• class MamdaDataException
  MAMDA data exceptions.

• class MamdaErrorListener
MamdaErrorListener defines an interface for handling error notifications for a MamdaSubscription.

- class MamdaBasicErrorListener
  MamdaBasicErrorListener defines an interface for handling error notifications for a MamdaBasicSubscription.

- class MamdaFields
- class MamdaFundamentalFields
- class MamdaFundamentalHandler
  MamdaFundamentalHandler is an interface for applications that want to have an easy way to access fundamental equity pricing/analysis attributes, indicators and ratios.

- class MamdaFundamentalListener
  MamdaFundamentalListener is a class that specializes in handling fundamental equity pricing/analysis attributes, indicators and ratios.

- class MamdaFundamentals
  MamdaFundamentals is an interface that provides access to the fundamental equity pricing/analysis attributes, indicators and ratios.

- class MamdaLock
- class MamdaMsgListener
  MamdaMsgListener defines an interface for handling MAMA messages for a MamdaSubscription.

- class MamdaBasicMsgListener
  MamdaBasicMsgListener defines an interface for handling MAMA messages for a MamdaBasicSubscription.

- class MamdaMultiParticipantHandler
  The MamdaMultiParticipantHandler class is an interface that allows a developer to be notified dynamically when participants are added to the list.

- class MamdaMultiParticipantManager
  MamdaMultiParticipantManager is a class that manages updates on a consolidated basis for securities that may be traded on multiple exchanges and which may have a national best bid and offer.

- class MamdaMultiSecurityHandler
  The MamdaMultiSecurityHandler class is an interface that allows a developer to be notified dynamically when securities are added to the list.
• class MamdaMultiSecurityManager
  
  *MamdaMultiSecurityManager* is a class that manages updates on an arbitrary number of securities that may be traded on multiple exchanges.

• class MamdaOrderImbalanceFields
  
  *Utility cache of MamaFieldDescriptors which are used internally by the API when accessing imbalance related fields from update messages.*

• class MamdaOrderImbalanceHandler
  
  *MamdaOrderImbalanceHandler* is an interface for applications that want to have an easy way to handle order imbalance updates.

• class MamdaOrderImbalanceListener
  
  A *MamdaOrderImbalanceListener* is a class that specializes in handling order imbalance updates.

• class MamdaOrderImbalanceRecap
• class MamdaOrderImbalanceSide
• class MamdaOrderImbalanceType
• class MamdaOrderImbalanceUpdate
• class MamdaPubStatus
  
  *MamdaPubStatus* is an interface that provides access to the Security Status fields such as symbol announce messages.

• class MamdaPubStatusFields
• class MamdaPubStatusHandler
  
  *MamdaPubStatusHandler* is an interface for applications that want to have an easy way to handle feed handler publisher status updates.

• class MamdaPubStatusListener
  
  *MamdaPubStatusListener* is a class that specializes in handling Publisher (Feed Handler) Status updates.

• class MamdaQualityListener
  
  *MamdaQualityListener* defines an interface for handling changes in quality notifications for a *MamdaSubscription*.

• class MamdaBasicQualityListener
  
  *MamdaBasicQualityListener* defines an interface for handling changes in quality notifications for a *MamdaBasicSubscription*.

• class MamdaQuery
• class MamdaOrQuery
• class MamdaAndQuery
• class MamdaEqualsQuery
• class MamdaDateQuery
• class MamdaContainsAllQuery
• class MamdaContainsQuery
• class MamdaQuoteChecker

  MamdaQuoteChecker is a class that provides quotes sanity checking by periodically requesting snapshots of the quotes from the publisher and comparing that with an quotes being maintained in real time.

• class MamdaQuoteClosing

  MamdaQuoteClosing is an interface that provides access to quote closing related fields.

• class MamdaQuoteFields

  Utility cache of MamaFieldDescriptors which are used internally by the API when accessing quote related fields from update messages.

• class MamdaQuoteGap

  MamdaQuoteGap is an interface that provides access to quote gap related fields.

• class MamdaQuoteHandler

  MamdaQuoteHandler is an interface for applications that want to have an easy way to handle quote updates.

• class MamdaQuoteListener

  MamdaQuoteListener is a class that specializes in handling quote updates.

• class MamdaQuoteOutOfSequence

  MamdaQuoteOutOfSequence is an interface that provides access to fields related to quote updates.

• class MamdaQuotePossiblyDuplicate

  MamdaQuotePossiblyDuplicate is an interface that provides access to fields related to quote updates which are possible duplicates of previous quote updates.

• class MamdaQuoteRecap

  MamdaQuoteRecap is an interface that provides access to quote related fields.

• class MamdaQuoteUpdate

  MamdaQuoteUpdate is an interface that provides access to fields related to quote updates.

• class MamdaSecStatus
MamdaSecStatus is an interface that provides access to the Security Status fields such as symbol announce messages.

- class MamdaSecStatusFields
  Utility cache of MamaFieldDescriptors which are used internally by the API when accessing security status related fields from update messages.

- class MamdaSecStatusHandler
  MamdaSecStatusHandler is an interface for applications that want to have an easy way to handle security status updates.

- class MamdaSecStatusListener
  MamdaSecurityStatusListener is a class that specializes in handling security status updates.

- class MamdaSecStatusRecap
  MamdaSecStatus is an interface that provides access to the Security Status fields such as symbol announce messages.

- class MamdaSecStatusSymbolSourceAdapter
  MamdaSecStatusSymbolSourceAdapter is a simple adapter class that can be added as a handler to the MamdaSecStatusListener turning it into a MamdaSymbolSource.

- class MamdaSubscription
  A MamdaSubscription is used to register interest in a particular symbol and source.

- class MamdaSymbolSourceEvent
  MamdaSymbolSourceEvent is an interface that provides access to a sourced symbol name.

- class MamdaSymbolSourceHandler
  MamdaSymbolSourceHandler is an interface for applications that want to have an easy way to handle newly sourced symbol events.

- class MamdaTradeCancelOrError
  MamdaTradeCancelOrError is an interface that provides access to trade cancellation related fields.

- class MamdaTradeChecker
  MamdaTradeChecker is a class that provides trades sanity checking by periodically requesting snapshots of the trades from the publisher and comparing that with an trades being maintained in real time.

- class MamdaTradeClosing
MamdaTradeClosing is an interface that provides access to trade closing related fields.

- class MamdaTradeCorrection
  MamdaTradeCorrection is an interface that provides access to trade correction related fields.

- class MamdaTradeFields
  Utility cache of MamaFieldDescriptors which are used internally by the API when accessing trade related fields from update messages.

- class MamdaTradeGap
  MamdaTradeGap is an interface that provides access to trade gap related fields.

- class MamdaTradeHandler
  MamdaTradeHandler is an interface for applications that want to have an easy way to handle trade updates.

- class MamdaTradeListener
  MamdaTradeListener is a class that specializes in handling trade updates.

- class MamdaTradeOutOfSequence
  MamdaTradeOutOfSequence is an interface that provides access to fields related to trade updates which have been identified as being out of sequence with previous update (e.g.

- class MamdaTradePossiblyDuplicate
  MamdaTradePossiblyDuplicate is an interface that provides access to fields related to trade updates which have been identified as being possible duplicates of previous updates.

- class MamdaTradeRecap
  MamdaTradeRecap is an interface that provides access to trade related fields.

- class MamdaTradeReport
  MamdaTradeReport is an interface that provides access to fields related to a trade report.

- class MamdaBookAtomicBookHandler
  MamdaBookAtomicBookHandler is an interface for applications that need to know when a MamdaBookAtomicListener finishes processing a single book update.

- class MamdaBookAtomicGap
**MamdaBookAtomicGap** is an interface that provides access to order book atomic update gap related fields.

- **class MamdaBookAtomicLevel**
  
  *MamdaBookAtomicLevel* is an interface that provides access to trade related fields.

- **class MamdaBookAtomicLevelEntry**
  
  *MamdaBookAtomicLevelEntry* is an interface that provides access to Price Level and Price Level Entry fields.

- **class MamdaBookAtomicLevelEntryHandler**
  
  *MamdaBookAtomicLevelEntryHandler* is an interface for applications that want to have an easy way to handle order book Price Level & Entry updates.

- **class MamdaBookAtomicLevelHandler**
  
  *MamdaBookAtomicLevelHandler* is an interface for applications that want to have an easy way to handle order book Price Level updates.

- **class MamdaBookAtomicListener**

  *MamdaBookAtomicListener* is a class that specializes in handling order book updates.

- **class MamdaOrderBook**

  *MamdaOrderBook* is a class that provides order book functionality, including iterators over price levels and entries within price levels.

- **class MamdaOrderBookBasicDelta**

  *MamdaOrderBookBasicDelta* is a class that saves information about a basic order book delta.

- **class MamdaOrderBookBasicDeltaList**

  *MamdaOrderBookBasicDeltaList* is a class that saves information about an order book delta that involves multiple entries and/or price levels.

- **class MamdaOrderBookChecker**

  *MamdaOrderBookChecker* is a class that provides order book sanity checking by periodically requesting snapshots of the order book from the publisher and comparing that with an order book being maintained in real time.

- **class MamdaOrderBookCheckerHandler**

  *MamdaOrderBookCheckerHandler* is an interface for applications that want to handle the results of the *MamdaOrderBookChecker*.

- **class MamdaOrderBookClear**

  [Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen]
MamdaOrderBookClear is an interface that provides access to order book related fields.

- class MamdaOrderBookComplexDelta
  
  MamdaOrderBookComplexDelta is a class that saves information about a complex order book delta.

- class MamdaOrderBookConcreteComplexDelta
  
  MamdaOrderBookConcreteComplexDelta is a class that saves information about a complex order book delta.

- class MamdaOrderBookConcreteSimpleDelta
  
  MamdaOrderBookConcreteSimpleDelta is a class that saves information about a simple order book delta.

- class MamdaOrderBookDelta
  
  MamdaOrderBookDelta is an interface that provides access to order book related fields.

- class MamdaOrderBookEntry
  
  MamdaOrderBookEntry is a class that represents an entry within a price level of an order book.

- class MamdaOrderBookEntryFilter

- class MamdaOrderBookEntryManager
  
  MamdaOrderBookEntryManager is a class that provides a global order book lookup service, matching entry IDs that are unique across a set of order books.

- class MamdaOrderBookException
  
  MamdaOrderBookException is generated when an inconsistent state is detected in an order book.

- class MamdaOrderBookDuplicateEntry
  
  MamdaOrderBookDuplicateEntry is generated when an existing entry is unexpectedly encountered when updating a MamdaOrderBook or MamdaOrderBookEntryManager.

- class MamdaOrderBookMissingEntry
  
  MamdaOrderBookMissingEntry is generated when an expected entry is not found when updating a MamdaOrderBook or MamdaOrderBookEntryManager.

- class MamdaOrderBookInvalidEntry
  
  MamdaOrderBookInvalidEntry is generated when an entry is applied as an update to or deletion from an order book but the internal references to price level and/or order book do not exist.
• class MamdaOrderBookFields
  Utility cache of MamaFieldDescriptors which are used internally by the API when accessing orderbook related fields from update messages.

• class MamdaOrderBookGap
  MamdaOrderBookGap is an interface that provides access to order book gap related fields.

• class MamdaOrderBookHandler
  MamdaOrderBookHandler is an interface for applications that want to have an easy way to handle order book updates.

• class MamdaOrderBookListener
  MamdaOrderBookListener is a class that specializes in handling order book updates.

• class MamdaOrderBookPriceLevel
  MamdaOrderBookPriceLevel is a class that provides a price level type for order books.

• class MamdaOrderBookRecap
  MamdaOrderBookRecap is an interface that provides access to order book related fields.

• class MamdaOrderBookSimpleDelta
  MamdaOrderBookSimpleDelta is a class that saves information about a simple order book delta.

• class MamdaOrderBookTypes
  MamdaOrderBookTypes is a class that provides order book related typed.

• class MamdaOptionChain
  MamdaOptionChain is a specialized class to represent market data option chains.

• class MamdaOptionChainHandler
  Subclasses of this interface can be registered with the MamdaOptionChainListener in order to receive callbacks whenever the state of the underlying option chain changes on receipt of options updates.

• class MamdaOptionChainListener
  MamdaOptionChainListener is a class that specializes in handling and managing option chain updates.

• class MamdaOptionChainView
5.1 Wombat Namespace Reference

A class that represents a "view" of a subset of an option chain.

- class **MamdaOptionChainViewRangeHandler**
  Class to handle change in a *MamdaOptionChainView* range.

- class **MamdaOptionContract**
  A class that represents a single option contract.

- struct **char_str_less_than**

- class **MamdaOptionContractSet**
  A class that represents a set of option contracts at a given strike price.

- class **MamdaOptionExchangeUtils**
  A class with static utility functions for dealing with exchanges.

- class **MamdaOptionExpirationDateSet**
  A class that represents a set of expiration dates, each of which contains a set of strike prices, each of which contains a set of option contracts, each of which contains exchange-specific contracts.

- class **MamdaOptionExpiration Strikes**
  A class that represents a set of strike prices at a particular expiration date.

- class **MamdaOptionFields**
  Utility cache of *MamaFieldDescriptors* which are used internally by the API when accessing options related fields from update messages.

- class **MamdaOptionSeriesUpdate**
  *MamdaOptionSeriesUpdate* is an interface that provides access to fields related to option series update events.

- class **MamdaOptionStrikeSet**
  A class that represents the call and put contract sets at a given strike price.

- class **MamdaNewsFields**

- class **MamdaNewsHeadline**
  *MamdaNewsHeadline* represents a news headline and includes information about many types of meta-data attributes associated with the headline.

- class **MamdaNewsHeadlineHandler**
  *MamdaNewsHandler* is an interface for applications that want to have an easy way to handle news headlines.
• class MamdaNewsManager
  
  `MamdaNewsManager` provides a class for managing access to streaming news headlines, headline queries, individual story queries, etc.

• class MamdaNewsMetaData
  
  `MamdaNewsMetaData` represents information about many types of attributes associated with the news headline.

• class MamdaNewsQueryHandler
  
  `MamdaNewsHandler` is an interface for applications that want to have an easy way to handle news queries.

• class MamdaNewsStory
  
  `MamdaNewsStory` represents a complete text of a news story.

• class MamdaNewsStoryHandler
  
  `MamdaNewsHandler` is an interface for applications that want to have an easy way to handle news stories.

**Typedefs**

• typedef set<double> StrikeSet
• typedef const char * MamdaNewsStoryId
• typedef const char * MamdaNewsHeadlineId

**Enumerations**

• enum MamdaCheckerType { MAMDA_CHECK_TYPE_NONE, MAMDA_CHECK_TYPE_SNAPSHOT, MAMDA_CHECK_TYPE_APPLY_DELTA }

• enum MamdaErrorSeverity { MAMDA_SEVERITY_OK, MAMDA_SEVERITY_LOW, MAMDA_SEVERITY_HIGH }

  `MAMDA error severities are intended to provide a hint to the application as to the severity of an error.`

• enum MamdaErrorCode {
  MAMDA_ERROR_NO_ERROR, MAMDA_ERROR_BAD_SYMBOL, MAMDA_ERROR_EXPIRED, MAMDA_ERROR_TIME_OUT, MAMDA_ERROR_ENTITLEMENT, MAMDA_ERROR_NOT_FOUND, MAMDA_ERROR_DELETE
  }

  `MAMDA error codes are currently a subset of the MAMA MsgStatus codes.`
• enum MamdaFieldState { MODIFIED = 2, NOT_MODIFIED = 1, NOT_INITIALIZED = 0 }

  An enumeration representing field state.

• enum MamdaSecurityStatus {
  SECURITY_STATUS_NONE, SECURITY_STATUS_NORMAL,
  SECURITY_STATUS_CLOSED, SECURITY_STATUS_HALTED,
  SECURITY_STATUS_NOT_EXIST, SECURITY_STATUS_DELETED,
  SECURITY_STATUS_AUCTION, SECURITY_STATUS_CROSSING,
  SECURITY_STATUS_SUSPENDED, SECURITY_STATUS_AT_LAST,
  SECURITY_STATUS_UNKNOWN = 99 }

  An enumeration representing the status of a security such as whether or not it is halted or closed for trading.

• enum MamdaSecurityStatusQual {
  SECURITY_STATUS_QUAL_NONE = 0, SECURITY_STATUS_QUAL_OPENING = 1, SECURITY_STATUS_QUAL_EXCUSED = 7, SECURITY_STATUS_QUAL_WITHDRAWN = 8,
  SECURITY_STATUS_QUAL_SUSPENDED = 9, SECURITY_STATUS_QUAL_RESUME = 11, SECURITY_STATUS_QUAL_QUOTE_RESUME = 12, SECURITY_STATUS_QUAL_TRADE_RESUME = 13,
  SECURITY_STATUS_QUAL_RESUME_TIME = 14, SECURITY_STATUS_QUAL_MKT_IMB_BUY = 16, SECURITY_STATUS_QUAL_MKT_IMB_SELL = 17, SECURITY_STATUS_QUAL_NO_MKT_IMB = 18,
  SECURITY_STATUS_QUAL_MOC_IMB_BUY = 19, SECURITY_STATUS_QUAL_MOC_IMB_SELL = 20, SECURITY_STATUS_QUAL_NO_MOC_IMB = 21, SECURITY_STATUS_QUAL_ORDER_IMB = 22,
  SECURITY_STATUS_QUAL_ORDER_INF = 23, SECURITY_STATUS_QUAL_ORDER_IMB_BUY = 24, SECURITY_STATUS_QUAL_ORDER_IMB_SELL = 25, SECURITY_STATUS_QUAL_ORDER_IMB_NONE = 26,
  SECURITY_STATUS_QUAL_LOA_IMBALANCE_BUY = 27, SECURITY_STATUS_QUAL_LOA_IMBALANCE_SELL = 28, SECURITY_STATUS_QUAL_NO_LOA_IMBALANCE = 29,
  SECURITY_STATUS_QUAL_ORDERS_ELIMINATED = 30,
  SECURITY_STATUS_QUAL_RANGE_ID = 31, SECURITY_STATUS_QUAL_RESERVED = 32, SECURITY_STATUS_QUAL_RESERVED = 33, SECURITY_STATUS_QUAL_FROZEN = 34,
  SECURITY_STATUS_QUAL_PREOPEN = 35, SECURITY_STATUS_QUAL_THO_IMBALANCE_BUY = 36, SECURITY_STATUS_QUAL_THO_IMBALANCE_SELL = 37, SECURITY_STATUS_QUAL_NO_THO_IMBALANCE = 38,
SECURITY_STATUS_QUAL_ADD_INFO = 41, SECURITY_STATUS_QUAL_IPO_IMBALANCE_BUY = 42, SECURITY_STATUS_QUAL_IPO_IMBALANCE_SELL = 43, SECURITY_STATUS_QUAL_NO_IPO_IMBALANCE = 44,
SECURITY_STATUS_QUAL_EMCP_IMB_BUY = 46, SECURITY_STATUS_QUAL_EMCP_IMB_SELL = 47, SECURITY_STATUS_QUAL_EMCP_IMB_NONE = 48, SECURITY_STATUS_QUAL_OPEN_DELAY = 51,
SECURITY_STATUS_QUAL_NO_OPEN_NO_RESUME = 52, SECURITY_STATUS_QUAL_PRICE_IND = 53, SECURITY_STATUS_QUAL_EQUIPMENT = 54, SECURITY_STATUS_QUAL_FILINGS = 55,
SECURITY_STATUS_QUAL_NEWS = 56, SECURITY_STATUS_QUAL_NEWS_DISSEM = 57, SECURITY_STATUS_QUAL_LISTING = 58, SECURITY_STATUS_QUAL_OPERATION = 59,
SECURITY_STATUS_QUAL_INFO = 60, SECURITY_STATUS_QUAL_SEC = 61, SECURITY_STATUS_QUAL_TIMES = 62, SECURITY_STATUS_QUAL_OTHER = 63,
SECURITY_STATUS_QUALRELATED = 64, SECURITY_STATUS_QUAL_IPO = 65, SECURITY_STATUS_QUAL_PRE_CROSS = 66, SECURITY_STATUS_QUAL_CROSS = 67,
SECURITY_STATUS_QUAL_RELEASED_FOR_QUOTATION = 68, SECURITY_STATUS_QUAL_IPO_WINDOW_EXT = 69, SECURITY_STATUS_QUAL_PRECLOSING = 70, SECURITY_STATUS_QUAL_AUCTION_EXTENSION = 71,
SECURITY_STATUS_QUAL_VOLATILITY_AUCTION = 72, SECURITY_STATUS_QUAL_SECURITY_AUTHOURISED = 73, SECURITY_STATUS_QUAL_SECURITY_FORBIDDEN = 74, SECURITY_STATUS_QUAL_FAST_MARKET = 75,
SECURITY_STATUS_QUAL_SLOW_MARKET = 76, SECURITY_STATUS_QUAL_SUB_PENNY_TRADING = 77, SECURITY_STATUS_QUAL_ORDER_INPUT = 78, SECURITY_STATUS_QUAL_PRE_ORDER_MATCHING = 79,
SECURITY_STATUS_QUAL_ORDER_MATCHING = 80, SECURITY_STATUS_QUAL_BLOCKING = 81, SECURITY_STATUS_QUAL_ORDER_CANCEL = 82, SECURITY_STATUS_QUAL_FIXED_PRICE = 83,
SECURITY_STATUS_QUAL_SALES_INPUT = 84, SECURITY_STATUS_QUAL_EXCHANGE_INTERVENTION = 85, SECURITY_STATUS_QUAL_PRE_AUCTION = 86, SECURITY_STATUS_QUAL_ADJUST = 87,
SECURITY_STATUS_QUAL_ADJUST_ON = 88, SECURITY_STATUS_QUAL_LATE_TRADING = 89, SECURITY_STATUS_QUAL_ENQUIRE = 90, SECURITY_STATUS_QUAL_PRE_NIGHT_TRADING = 91,

An enumeration representing the status of a security such as whether or not it is halted or closed for trading.

• enum MamdaTradeDirection {
  TRADE_DIR_ZERO, TRADE_DIR_PLUS, TRADE_DIR_MINUS, TRADE_DIR_ZERO_PLUS,
  TRADE_DIR_ZERO_MINUS, TRADE_DIR_NA, TRADE_DIR_UNKNOWN = 99 }

  An enumeration representing trade tick direction, relative to the previous "last" trade.

• enum MamdaTradeExecVenue {
TRADE_EXEC_VENUE_UNKNOWN, TRADE_EXEC_VENUE_ON_EXCHANGE, TRADE_EXEC_VENUE_ON_EXCHANGE_OFF_BOOK, TRADE_EXEC_VENUE_OFF_EXCHANGE, TRADE_EXEC_VENUE_SYSTEM_INTERNALISER, TRADE_EXEC_VENUE_ON_EXCHANGE_DARK_BOOK, TRADE_EXEC_VENUE_ON_EXCHANGE_ON_BOOK }

An enumeration representing trade execution venue.

• enum MamdaTradeSide { TRADE_SIDE_UNKNOWN = 0, TRADE_SIDE_BUY = 1, TRADE_SIDE_SELL = 2 }

An enumeration representing trade side.

• enum MamdaUncrossPriceInd { UNCROSS_NONE, UNCROSS_INDICATIVE, UNCROSS_FIRM, UNCROSS_INSUFFICIENT_VOL }

An enumeration representing the uncross price Ind.

• enum MamdaOrderBookCheckType { MAMDA_BOOK_CHECK_TYPE_NONE = 0, MAMDA_BOOK_CHECK_TYPE_SNAPSHOT = 1, MAMDA_BOOK_CHECK_TYPE_APPLY_DELTA = 2 }

• enum MamdaOptionAtTheMoneyCompareType { MAMDA_AT_THE_MONEY_COMPARE_MID_QUOTE = 0, MAMDA_AT_THE_MONEY_COMPARE_BID = 1, MAMDA_AT_THE_MONEY_COMPARE_ASK = 2, MAMDA_AT_THE_MONEY_COMPARE_LAST_TRADE = 3 }

• enum MamdaOptionPutCall { MAMDA_PUT_CALL_CALL = 'C', MAMDA_PUT_CALL_PUT = 'P', MAMDA_PUT_CALL_UNKNOWN = 'Z' }

Enumeration for indicating whether an option contract is a put or a call.

• enum MamdaOptionExerciseStyle { MAMDA_EXERCISE_STYLE_AMERICAN = 'A', MAMDA_EXERCISE_STYLE_EUROPEAN = 'E', MAMDA_EXERCISE_STYLE_CAPPED = 'C', MAMDA_EXERCISE_STYLE_UNKNOW = 'Z' }

Enumeration for indicating the style of an individual option contract.

• enum MamdaNewsQueryType { QUERY_TYPE_UNKNOWN = 0, QUERY_TYPE_HISTORICAL = 1, QUERY_TYPE_SUBSCRIPTION = 2, QUERY_TYPE_HISTORICAL_SUBSCRIPTION = 3 }

• enum MamdaNewsPriority { MAMDA_NEWS_PRIORITY_NONE = 0, MAMDA_NEWS_PRIORITY_NORMAL = 5, MAMDA_NEWS_PRIORITY_HOT = 9 }

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
5.1 Wombat Namespace Reference

Functions

- MAMAExpDLL const char * mamdaCheckTypeToString (MamdaCheckerType type)
- MAMDAExpDLL const char * toString (MamdaFieldState fieldState)
  
  Convert a MamdaFieldState to an appropriate, displayable string.

- MAMDAExpDLL const char * toString (MamdaSecurityStatus securityStatus)
  
  Convert a MamdaSecurityStatus to an appropriate, displayable string.

- MAMDAExpDLL MamdaSecurityStatus mamdaSecurityStatusFromString (const char *securityStatusStr)
  
  Convert a string representation of a security status to the enumeration.

- MAMDAExpDLL const char * toString (MamdaSecurityStatusQual securityStatusQual)
  
  Convert a MamdaSecurityStatusQual to an appropriate, displayable string.

- MAMDAExpDLL MamdaSecurityStatusQual mamdaSecurityStatusQualFromString (const char *securityStatusQualStr)
  
  Convert a string representation of a security status to the enumeration.

- MAMDAExpDLL const char * toString (MamdaTradeDirection tradeDir)
  
  Convert a MamdaTradeDirection to an appropriate, displayable string.

- MAMDAExpDLL MamdaTradeDirection mamdaTradeDirectionFromString (const char *tradeDirStr)
  
  Convert a string representation of a trade direction to the enumeration.

- MAMDAExpDLL const char * toString (MamdaTradeExecVenue tradeExecVenue)
  
  Convert a MamdaTradeExecVenue to an appropriate, displayable string.

- MAMDAExpDLL MamdaTradeExecVenue mamdaTradeExecVenueFromString (const char *tradeExecVenueStr)
  
  Convert a string representation of a trade execution venue to the enumeration.

- MAMDAExpDLL const char * toString (MamdaTradeSide tradeSide)
  
  Convert a MamdaTradeSide to an appropriate, displayable string.

- MAMDAExpDLL MamdaTradeSide mamdaTradeSideFromString (const char *tradeSideStr)
Convert a string representation of a trade side to the enumeration.

- MAMDAExpDLL const char * toString (MamdaUncrossPriceInd security-Status)
  
  Convert a MamdaUncrossPriceInd to an appropriate, displayable string.

- MAMDAExpDLL MamdaUncrossPriceInd mamdaUncrossPriceIndFromString (const char *uncrossPriceInd)
  
  Convert a string representation of a uncross price Ind to the enumeration.

- MAMDAExpDLL const char * getMamdaVersion (void)
  
  Get the version of Mamda.

- MAMAExpDLL const char * mamdaOrderBookCheckTypeToString (Mamda-OrderBookCheckType type)

5.1.1 Detailed Description

MamdaOrderImbalanceUpdate is an interface that provides access to order imbalance related fields.

5.1.2 Typedef Documentation

5.1.2.1 typedef set<double> Wombat::StrikeSet

5.1.2.2 typedef const char * Wombat::MamdaNewsStoryId

5.1.2.3 typedef const char * Wombat::MamdaNewsHeadlineId

5.1.3 Enumeration Type Documentation

5.1.3.1 enum Wombat::MamdaCheckerType

Enumerator:

MAMDA_CHECK_TYPE_NONE
MAMDA_CHECK_TYPE_SNAPSHOT
MAMDA_CHECK_TYPE_APPLY_DELTA

```cpp
31 { 32 MAMDA_CHECK_TYPE_NONE, 33 MAMDA_CHECK_TYPE_SNAPSHOT, 34 MAMDA_CHECK_TYPE_APPLY_DELTA 35 };
```
5.1.3.2 enum Wombat::MamdaErrorSeverity

MAMDA error severities are intended to provide a hint to the application as to the severity of an error.

In relation to a MamdaSubscription, a MAMDA_SEVERITY_HIGH means that the subscription has been deactivated and a MAMDA_SEVERITY_LOW means that the condition may be temporary.

Enumerator:

MAMDA_SEVERITY_OK
MAMDA_SEVERITY_LOW
MAMDA_SEVERITY_HIGH

5.1.3.3 enum Wombat::MamdaErrorCode

MAMDA error codes are currently a subset of the MAMA MsgStatus codes.

Enumerator:

MAMDA_ERROR_NO_ERROR
MAMDA_ERROR_BAD_SYMBOL
MAMDA_ERROR_EXPIRED
MAMDA_ERROR_TIME_OUT
MAMDA_ERROR_ENTITLEMENT
MAMDA_ERROR_NOT_FOUND
MAMDA_ERROR_DELETE

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
5.1.3.4 enum Wombat::MamdaFieldState

An enumeration representing field state.

MODIFIED value indicates the field was updated in last tick, NOT_MODIFIED value indicates that there was no change in the last tick, NOT_INITIALISED value indicates that the field has never been updated.

Enumerator:

\[\begin{align*}
&\text{MODIFIED} = 2, \\
&\text{NOT_MODIFIED} = 1, \\
&\text{NOT_INITIALISED} = 0
\end{align*}\]

5.1.3.5 enum Wombat::MamdaSecurityStatus

An enumeration representing the status of a security such as whether or not it is halted or closed for trading.

Enumerator:

\[\begin{align*}
&\text{SECURITY_STATUS_NONE}, \quad \text{toString()} \text{ returns "None"} \\
&\text{SECURITY_STATUS_NORMAL}, \quad \text{toString()} \text{ returns "Normal"} \\
&\text{SECURITY_STATUS_CLOSED} \\
&\text{SECURITY_STATUS_HALTED} \\
&\text{SECURITY_STATUS_NOT_EXIST} \\
&\text{SECURITY_STATUS_DELETED} \\
&\text{SECURITY_STATUS_AUCTION} \\
&\text{SECURITY_STATUS_CROSSING} \\
&\text{SECURITY_STATUS_SUSPENDED} \\
&\text{SECURITY_STATUS_AT_LAST} \\
&\text{SECURITY_STATUS_UNKNOWN}
\end{align*}\]
5.1 Wombat Namespace Reference

39 SECURITY_STATUS_HALTED, /* toString() returns "Halted" */
40 SECURITY_STATUS_NOT_EXIST, /* toString() returns "NotExist" */
41 SECURITY_STATUS_DELETED, /* toString() returns "Deleted" */
42 SECURITY_STATUS_AUCTION, /* toString() returns "Auction" */
43 SECURITY_STATUS_CROSSING, /* toString() returns "Crossing" */
44 SECURITY_STATUS_SUSPENDED, /* toString() returns "Suspended" */
45 SECURITY_STATUS_AT_LAST, /* toString() returns "AtLast" */
46 SECURITY_STATUS_UNKNOWN = 99 /* toString() returns "Unknown" */
47 

5.1.3.6 enum Wombat::MamdaSecurityStatusQual

An enumeration representing the status of a security such as whether or not it is halted or closed for trading.

Enumerator:

SECURITY_STATUS_QUAL_NONE
SECURITY_STATUS_QUAL_OPENING
SECURITY_STATUS_QUAL_EXCUSED
SECURITY_STATUS_QUAL_WITHDRAWN
SECURITY_STATUS_QUAL_SUSPENDED
SECURITY_STATUS_QUAL_RESUME
SECURITY_STATUS_QUAL_QUOTE_RESUME
SECURITY_STATUS_QUAL_TRADE_RESUME
SECURITY_STATUS_QUAL_RESUME_TIME
SECURITY_STATUS_QUAL_MKT_IMB_BUY
SECURITY_STATUS_QUAL_MKT_IMB_SELL
SECURITY_STATUS_QUAL_NO_MKT_IMB
SECURITY_STATUS_QUAL_MOC_IMB_BUY
SECURITY_STATUS_QUAL_MOC_IMB_SELL
SECURITY_STATUS_QUAL_NO_MOC_IMB
SECURITY_STATUS_QUAL_ORDER_IMB
SECURITY_STATUS_QUAL_ORDER_INF
SECURITY_STATUS_QUAL_ORDER_IMB_BUY
SECURITY_STATUS_QUAL_ORDER_IMB_SELL
SECURITY_STATUS_QUAL_ORDER_IMB_NONE
SECURITY_STATUS_QUAL_LOA_IMBALANCE_BUY
SECURITY_STATUS_QUAL_LOA_IMBALANCE_SELL
SECURITY_STATUS_QUAL_NO_LOA_IMBALANCE
SECURITY_STATUS_QUAL_ORDERS_ELIMINATED
SECURITY_STATUS_QUAL_RANGE_ID
SECURITY_STATUS_QUAL_ITS_PREOPEN
SECURITY_STATUS_QUAL_RESERVED
SECURITY_STATUS_QUAL_FROZEN
SECURITY_STATUS_QUAL_PREOPEN
SECURITY_STATUS_QUAL_THO_IMBALANCE_BUY
SECURITY_STATUS_QUAL_THO_IMBALANCE_SELL
SECURITY_STATUS_QUAL_NO_THO_IMBALANCE
SECURITY_STATUS_QUAL_ADD_INFO
SECURITY_STATUS_QUAL_IPO_IMBALANCE_BUY
SECURITY_STATUS_QUAL_IPO_IMBALANCE_SELL
SECURITY_STATUS_QUAL_NO_IPO_IMBALANCE
SECURITY_STATUS_QUAL_EMCI_MBC_BUY
SECURITY_STATUS_QUAL_EMCI_MBC_SELL
SECURITY_STATUS_QUAL_EMCI_MBC_NONE
SECURITY_STATUS_QUAL_OPEN_DELAY
SECURITY_STATUS_QUAL_NO_OPEN_NO_RESUME
SECURITY_STATUS_QUAL_PRICE_IND
SECURITY_STATUS_QUAL_EQUIPMENT
SECURITY_STATUS_QUAL_FILINGS
SECURITY_STATUS_QUAL_NEWS
SECURITY_STATUS_QUAL_NEWS_DISSEM
SECURITY_STATUS_QUAL_LISTING
SECURITY_STATUS_QUAL_OPERATION
SECURITY_STATUS_QUAL_INFO
SECURITY_STATUS_QUAL_SEC
SECURITY_STATUS_QUAL_TIMES
SECURITY_STATUS_QUAL_OTHER
SECURITY_STATUS_QUAL_RELATED
SECURITY_STATUS_QUAL_IPO
SECURITY_STATUS_QUAL_PRE_CROSS
SECURITY_STATUS_QUAL_CROSS
SECURITY_STATUS_QUAL_RELEASED_FOR_QUOTATION
SECURITY_STATUS_QUAL_IPO_WINDOW_EXT
SECURITY_STATUS_QUAL_PRECLOSING
SECURITY_STATUS_QUAL_AUCTION_EXTENSION
SECURITY_STATUS_QUAL_VOLATILITY_AUCTION
SECURITY_STATUS_QUAL_SECURITY_AUTHORISED
SECURITY_STATUS_QUAL_SECURITY_FORBIDDEN
SECURITY_STATUS_QUAL_FAST_MARKET
SECURITY_STATUS_QUAL_SLOW_MARKET
SECURITY_STATUS_QUAL_SUB_PENNY_TRADING
SECURITY_STATUS_QUAL_ORDER_INPUT
SECURITY_STATUS_QUAL_PRE_ORDER_MATCHING
SECURITY_STATUS_QUAL_ORDER_MATCHING
SECURITY_STATUS_QUAL_BLOCKING
SECURITY_STATUS_QUAL_ORDER_CANCEL
SECURITY_STATUS_QUAL_FIXED_PRICE
SECURITY_STATUS_QUAL_SALES_INPUT
SECURITY_STATUS_QUAL.Exchange_INTERVENTION
SECURITY_STATUS_QUAL_PRE_AUCTION
SECURITY_STATUS_QUAL_ADJUST
SECURITY_STATUS_QUAL_ADJUST_ON
SECURITY_STATUS_QUAL_LATE_TRADING
SECURITY_STATUS_QUAL_ENQUIRE
SECURITY_STATUS_QUAL_PRE_NIGHT_TRADING
SECURITY_STATUS_QUAL_OPEN_NIGHT_TRADING
SECURITY_STATUS_QUAL_SUB_HOLIDAY
SECURITY_STATUS_QUAL_BID_ONLY
SECURITY_STATUS_QUAL_ASK_ONLY
SECURITY_STATUS_QUAL_UNKNOWN
SECURITY_STATUS_QUAL.OPENING_DELAY_COMMON
SECURITY_STATUS_QUAL_RESUME_COMMON
SECURITY_STATUS_QUAL_NO_OPEN_NO_RESUME_COMMON
SECURITY_STATUS_QUAL_NEWS_DISSEMINATIONRELATED
SECURITY_STATUS_QUAL_ORDER_INFLUXRELATED
SECURITY_STATUS_QUAL_ORDER_IMBALANCERELATED
SECURITY_STATUS_QUAL_INFORMATION_REQUESTED.RELATED
SECURITY_STATUS_QUAL_NEWS_PENDING_RELATED
SECURITY_STATUS_QUAL_EQUIPMENT_CHANGEOVER_RELATED
SECURITY_STATUS_QUAL_SUB_PENNY_TRADING_RELATED
SECURITY_STATUS_QUAL_IMBALANCE_PREOPEN_BUY
SECURITY_STATUS_QUAL_IMBALANCE_PREOPEN_SELL
SECURITY_STATUS_QUAL_IMBALANCE_PRECLOSE_BUY
SECURITY_STATUS_QUAL_IMBALANCE_PRECLOSE_SELL
SECURITY_STATUS_QUAL_VOLATILITY_PAUSE
SECURITY_STATUS_QUAL_VOLATILITY_PAUSE_QUOTE_RESUME
SECURITY_STATUS_QUAL_VOLATILITY_GUARD
SECURITY_STATUS_QUAL_VOLATILITY_GUARD_QUOTE_RESUME
SECURITY_STATUS_QUAL_LIMIT_STATE_BID
SECURITY_STATUS_QUAL_LIMIT_STATE_ASK
SECURITY_STATUS_QUAL_LIMITSTATEBID_LIMITSTATEASK
SECURITY_STATUS_QUAL_CIRCUIT_BREAKER_LEVEL_1
SECURITY_STATUS_QUAL_CIRCUIT_BREAKER_LEVEL_2
SECURITY_STATUS_QUAL_CIRCUIT_BREAKER_LEVEL_3

35 {
36   SECURITY_STATUS_QUAL_NONE = 0, // toString() - None
37   SECURITY_STATUS_QUAL_OPENING = 1, // toString() - Opening
38   SECURITY_STATUS_QUAL_EXCUSED = 7, // toString() - Excused
39   SECURITY_STATUS_QUAL_WITHDRAWN = 8, // toString() - Withdrawn
40   SECURITY_STATUS_QUAL_SUSPENDED = 9, // toString() - Suspended
41   SECURITY_STATUS_QUAL_RESUME = 11, // toString() - Resume
42   SECURITY_STATUS_QUAL_QUOTE_RESUME = 12, // toString() - QuoteResume
43   SECURITY_STATUS_QUAL_TRADE_RESUME = 13, // toString() - TradeResume
44   SECURITY_STATUS_QUAL_RESUME_TIME = 14, // toString() - ResumeTime
45   SECURITY_STATUS_QUAL_MKT_IMB_BUY = 16, // toString() - MktImbBuy
46   SECURITY_STATUS_QUAL_MKT_IMB_SELL = 17, // toString() - MktImbSell
47   SECURITY_STATUS_QUAL_NO_MKT_IMB = 18, // toString() - NoMktImb
48   SECURITY_STATUS_QUAL_MOC_IMB_BUY = 19, // toString() - MocImbBuy
49   SECURITY_STATUS_QUAL_MOC_IMB_SELL = 20, // toString() - MocImbSell
50   SECURITY_STATUS_QUAL_NO_MOC_IMB = 21, // toString() - NoMocImb
51   SECURITY_STATUS_QUAL_ORDER_IMB = 22, // toString() - OrderImb
52   SECURITY_STATUS_QUAL_ORDER_INF = 23, // toString() - OrderInf
53   SECURITY_STATUS_QUAL_ORDER_IMB_BUY = 24, // toString() - OrderImbBuy
54   SECURITY_STATUS_QUAL_ORDER_IMB_SELL = 25, // toString() - OrderImbSell
55   SECURITY_STATUS_QUAL_ORDER_IMB_NONE = 26, // toString() - OrderImbNone
56   SECURITY_STATUS_QUAL_LOA_IMBALANCE_BUY = 27, // toString() - LoaImbBuy
57   SECURITY_STATUS_QUAL_LOA_IMBALANCE_SELL = 28, // toString() - LoaImbSell
58   SECURITY_STATUS_QUAL_NO_LOA_IMBALANCE = 29, // toString() - LoaNoImb
59   SECURITY_STATUS_QUAL_ORDERS_ELIMINATED = 30, // toString() - OrdersEliminated
60   SECURITY_STATUS_QUAL_RANGE_ID = 31, // toString() - RangeId
61   SECURITY_STATUS_QUAL_ITS_PREOPEN = 32, // toString() - ItsPreOpen
62   SECURITY_STATUS_QUAL_RESERVED = 33, // toString() - Reserved
<table>
<thead>
<tr>
<th></th>
<th>SECURITY_STATUS_QUAL_FROZEN</th>
<th>34, // toString() - Frozen</th>
</tr>
</thead>
<tbody>
<tr>
<td>63</td>
<td>SECURITY_STATUS_QUAL_PREOPEN</td>
<td>35, // toString() - PreOpen</td>
</tr>
<tr>
<td>64</td>
<td>SECURITY_STATUS_QUAL_THO_IMBALANCE_BUY</td>
<td>36, // toString() - ThoImbBuy</td>
</tr>
<tr>
<td>65</td>
<td>SECURITY_STATUS_QUAL_THO_IMBALANCE_SELL</td>
<td>37, // toString() - ThoImbSell</td>
</tr>
<tr>
<td>66</td>
<td>SECURITY_STATUS_QUAL_NO_THO_IMBALANCE</td>
<td>38, // toString() - ThoNoImb</td>
</tr>
<tr>
<td>67</td>
<td>SECURITY_STATUS_QUAL_ADD_INFO</td>
<td>41, // toString() - AddInfo</td>
</tr>
<tr>
<td>68</td>
<td>SECURITY_STATUS_QUAL_IPO_IMBALANCE_BUY</td>
<td>42, // toString() - IpoImbBuy</td>
</tr>
<tr>
<td>69</td>
<td>SECURITY_STATUS_QUAL_IPO_IMBALANCE_SELL</td>
<td>43, // toString() - IpoImbSell</td>
</tr>
<tr>
<td>70</td>
<td>SECURITY_STATUS_QUAL_IPO_IMBALANCE</td>
<td>44, // toString() - IpoNoImb</td>
</tr>
<tr>
<td>71</td>
<td>SECURITY_STATUS_QUAL_EMC_IMB_BUY</td>
<td>46, // toString() - EmcImbBuy</td>
</tr>
<tr>
<td>72</td>
<td>SECURITY_STATUS_QUAL_EMC_IMB_SELL</td>
<td>47, // toString() - EmcImbSell</td>
</tr>
<tr>
<td>73</td>
<td>SECURITY_STATUS_QUAL_EMC_IMB_NONE</td>
<td>48, // toString() - EmcImbNone</td>
</tr>
<tr>
<td>74</td>
<td>SECURITY_STATUS_QUAL_OPEN DELAY</td>
<td>51, // toString() - OpenDelay</td>
</tr>
<tr>
<td>75</td>
<td>SECURITY_STATUS_QUAL_NO_OPEN_NO_RESUME</td>
<td>52, // toString() - NoOpenNoResume</td>
</tr>
<tr>
<td>76</td>
<td>SECURITY_STATUS_QUAL_PRICE_IND</td>
<td>53, // toString() - PriceInd</td>
</tr>
<tr>
<td>77</td>
<td>SECURITY_STATUS_QUAL_EQUIPMENT</td>
<td>54, // toString() - Equipment</td>
</tr>
<tr>
<td>78</td>
<td>SECURITY_STATUS_QUAL_FILINGS</td>
<td>55, // toString() - Filings</td>
</tr>
<tr>
<td>79</td>
<td>SECURITY_STATUS_QUAL_NEWS</td>
<td>56, // toString() - News</td>
</tr>
<tr>
<td>80</td>
<td>SECURITY_STATUS_QUAL_NEWS_DISSEM</td>
<td>57, // toString() - NewsDissem</td>
</tr>
<tr>
<td>81</td>
<td>SECURITY_STATUS_QUAL_LISTING</td>
<td>58, // toString() - Listing</td>
</tr>
<tr>
<td>82</td>
<td>SECURITY_STATUS_QUAL_OPERATION</td>
<td>59, // toString() - Operation</td>
</tr>
<tr>
<td>83</td>
<td>SECURITY_STATUS_QUAL_INFO</td>
<td>60, // toString() - Info</td>
</tr>
<tr>
<td>84</td>
<td>SECURITY_STATUS_QUAL_SEC</td>
<td>61, // toString() - SEC</td>
</tr>
<tr>
<td>85</td>
<td>SECURITY_STATUS_QUAL_TIMES</td>
<td>62, // toString() - Times</td>
</tr>
<tr>
<td>86</td>
<td>SECURITY_STATUS_QUAL_OTHER</td>
<td>63, // toString() - Other</td>
</tr>
<tr>
<td>87</td>
<td>SECURITY_STATUS_QUAL_SECURITY_AUTHORISED</td>
<td>64, // toString() - Related</td>
</tr>
<tr>
<td>88</td>
<td>SECURITY_STATUS_QUAL_SECURITY_FORBIDDEN</td>
<td>65, // toString() - IPO</td>
</tr>
<tr>
<td>89</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>66, // toString() - Pre-Cross</td>
</tr>
<tr>
<td>90</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>67, // toString() - Cross</td>
</tr>
<tr>
<td>91</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>68, // toString() - Released For Quotation (IPO)</td>
</tr>
<tr>
<td>92</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>69, // toString() - IPO Window Extension</td>
</tr>
<tr>
<td>93</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>70, // toString() - PreClosing</td>
</tr>
<tr>
<td>94</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>71, // toString() - Auction Extension</td>
</tr>
<tr>
<td>95</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>72, // toString() - Volatility Auction</td>
</tr>
<tr>
<td>96</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>73, // toString() - SecurityAuthorized</td>
</tr>
<tr>
<td>97</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>74, // toString() - SecurityForbidden</td>
</tr>
<tr>
<td>98</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>75, // toString() - FastMarket</td>
</tr>
<tr>
<td>99</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>76, // toString() - SlowMarket</td>
</tr>
<tr>
<td>100</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>77, // toString() - SubPennyTrading</td>
</tr>
<tr>
<td>101</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>78, // toString() - OrderInput</td>
</tr>
<tr>
<td>102</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>79, // toString() - PreOrderMatching</td>
</tr>
<tr>
<td>103</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>80, // toString() - OrderMatching</td>
</tr>
<tr>
<td>104</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>81, // toString() - Blocking</td>
</tr>
<tr>
<td>105</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>82, // toString() - OrderCancel</td>
</tr>
<tr>
<td>106</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>83, // toString() - FixedPrice</td>
</tr>
<tr>
<td>107</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>84, // toString() - SalesInput</td>
</tr>
<tr>
<td>108</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>85, // toString() - ExchangeIntervention</td>
</tr>
<tr>
<td>109</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>86, // toString() - PreAuction</td>
</tr>
<tr>
<td>110</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>87, // toString() - Adjust</td>
</tr>
<tr>
<td>111</td>
<td>SECURITY_STATUSQUAL_SECURITY_WINDOW</td>
<td>88, // toString() - AdjustOn</td>
</tr>
<tr>
<td>112</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>89, // toString() - PreNightTrading</td>
</tr>
<tr>
<td>113</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>90, // toString() - Operation</td>
</tr>
<tr>
<td>114</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>91, // toString() - OpenNightTrading</td>
</tr>
<tr>
<td>115</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>92, // toString() - SubHoliday</td>
</tr>
<tr>
<td>116</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>93, // toString() - BidOnly</td>
</tr>
<tr>
<td>117</td>
<td>SECURITY_STATUS_QUAL_SECURITY_WINDOW</td>
<td>94, // toString() - AskOnly</td>
</tr>
</tbody>
</table>

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
SECURITY_STATUS_QUAL_UNKNOWN = 99, // toString() - Unknown
SECURITY_STATUS_QUAL_OPENING_DELAY_COMMON = 101, // toString() - OpenDelay
SECURITY_STATUS_QUAL_RESUME_COMMON = 102, // toString() - ResumeCO
SECURITY_STATUS_QUAL_NO_OPEN_NO_RESUME_COMMON = 103, // toString() - NoOpenNoResume
SECURITY_STATUS_QUAL_NEWS_DISSEMINATIONRELATED = 104, // toString() - NewsDisseminationRelated
SECURITY_STATUS_QUAL_ORDER_INFLUX_RELATED = 105, // toString() - OrderInfluxRelated
SECURITY_STATUS_QUAL_ORDER_IMMUTABLE_RELATED = 106, // toString() - OrderImmutableRelated
SECURITY_STATUS_QUAL_INFORMATION_REQUESTED_RELATED = 107, // toString() - InfoRequestRelated
SECURITY_STATUS_QUAL_NEWS_PENDING_RELATED = 108, // toString() - NewsPendingRelated
SECURITY_STATUS_QUAL_EQUIPMENT_CHANGEOVER_RELATED = 109, // toString() - EquipmentChangeoverRelated
SECURITY_STATUS_QUAL_SUB_PENNY_TRADING_RELATED = 110, // toString() - SubPennyTradingRelated
SECURITY_STATUS_QUAL_IMBALANCE_PREOPEN_BUY = 112, // toString() - ImbPreOpenBuy
SECURITY_STATUS_QUAL_IMBALANCE_PREOPEN_SELL = 113, // toString() - ImbPreOpenSell
SECURITY_STATUS_QUAL_IMBALANCE_PRECLOSE_BUY = 114, // toString() - ImbPreCloseBuy
SECURITY_STATUS_QUAL_IMBALANCE_PRECLOSE_SELL = 115, // toString() - ImbPreCloseSell
SECURITY_STATUS_QUAL_VOLATILITY_PAUSE = 116, // toString() - VolatilityPause
SECURITY_STATUS_QUAL_VOLATILITY_PAUSE_QUOTE_RESUME = 117, // toString() - VolatilityPauseResume
SECURITY_STATUS_QUAL_VOLATILITY_GUARD = 118, // toString() - VolatilityGuard
SECURITY_STATUS_QUAL_VOLATILITY_GUARD_QUOTE_RESUME = 119, // toString() - VolatilityGuardResume
SECURITY_STATUS_QUAL_LIMIT_STATE_BID = 120, // toString() - LimitStateBid
SECURITY_STATUS_QUAL_LIMIT_STATE_ASK = 121, // toString() - LimitStateAsk
SECURITY_STATUS_QUAL_LIMITSTATEBID_LIMITSTATEASK = 122, // toString() - LimitStateBidLimitStateAsk
SECURITY_STATUS_QUAL_CIRCUIT_BREAKER_LEVEL_1 = 123, // toString() - CircuitBreakerLevel1
SECURITY_STATUS_QUAL_CIRCUIT_BREAKER_LEVEL_2 = 124, // toString() - CircuitBreakerLevel2
SECURITY_STATUS_QUAL_CIRCUIT_BREAKER_LEVEL_3 = 125, // toString() - CircuitBreakerLevel3

5.1.3.7 enum Wombat::MamdaTradeDirection

An enumeration representing trade tick direction, relative to the previous "last" trade.

The TRADE_DIR_ZERO_PLUS and TRADE_DIR_ZERO_MINUS values indicate that there was no change in tick direction, but the previous non-zero change (some number of ticks ago) was up or down, respectively.

Enumerator:

- TRADE_DIR_ZERO
- TRADE_DIR_PLUS
- TRADE_DIR_MINUS
- TRADE_DIR_ZERO_PLUS
- TRADE_DIR_ZERO_MINUS
- TRADE_DIR_NA
- TRADE_DIR_UNKNOWN
5.1 Wombat Namespace Reference

```
38 {
39   TRADE_DIR_ZERO, /* toString() returns "" */
40   TRADE_DIR_PLUS, /* toString() returns "+" */
41   TRADE_DIR_MINUS, /* toString() returns "-" */
42   TRADE_DIR_ZERO_PLUS, /* toString() returns "0+" */
43   TRADE_DIR_ZERO_MINUS, /* toString() returns "0-" */
44   TRADE_DIR_NA, /* toString() returns "NA" */
45   TRADE_DIR UNKNOWN = 99 /* toString() returns "ZZ" */
46 }
```

5.1.3.8 enum Wombat::MamdaTradeExecVenue

An enumeration representing trade execution venue.

The TRADE_DIR ZERO PLUS and TRADE_DIR ZERO_MINUS values indicate that there was no change in tick direction, but the previous non-zero change (some number of ticks ago) was up or down, respectively.

Enumerator:

```
TRADE EXEC VENUE_UNKNOWN
TRADE EXEC VENUE ON EXCHANGE
TRADE EXEC VENUE ON EXCHANGE OFF BOOK
TRADE EXEC VENUE OFF EXCHANGE
TRADE EXEC VENUE SYSTEM INTERNALISER
TRADE EXEC VENUE ON EXCHANGE DARK BOOK
TRADE EXEC VENUE ON EXCHANGE ON BOOK
```

```
37 {
38   TRADE EXEC VENUE UNKNOWN, /* toString() returns "" */
39   TRADE EXEC VENUE ON EXCHANGE, /* toString() returns "+" */
40   TRADE EXEC VENUE ON EXCHANGE OFF BOOK, /* toString() returns "-" */
41   TRADE EXEC VENUE OFF EXCHANGE, /* toString() returns "0+" */
42   TRADE EXEC VENUE SYSTEM INTERNALISER,
43   TRADE EXEC VENUE ON EXCHANGE DARK BOOK,
44   TRADE EXEC VENUE ON EXCHANGE ON BOOK
45 }
```

5.1.3.9 enum Wombat::MamdaTradeSide

An enumeration representing trade side.

The value 0 maps to Unknown. The value 1 maps to Buy. The value 2 maps to Sell.

Enumerator:

```
TRADE SIDE UNKNOWN
```
TRADE_SIDE_BUY

TRADE_SIDE_SELL

37
38 TRADE_SIDE_UNKNOWN = 0,
39 TRADE_SIDE_BUY = 1,
40 TRADE_SIDE_SELL = 2
41 ;

5.1.3.10 enum Wombat::MamdaUncrossPriceInd

An enumeration representing the uncross price Ind.

Enumerator:

UNCROSS_NONE

UNCROSS_INDICATIVE

UNCROSS_FIRM

UNCROSS_INSUFFICIENT_VOL

34
35 { UNCROSS_NONE,
36 UNCROSS_INDICATIVE,
37 UNCROSS_FIRM,
38 UNCROSS_INSUFFICIENT_VOL
39 };

5.1.3.11 enum Wombat::MamdaOrderBookCheckType

Enumerator:

MAMDA_BOOK_CHECK_TYPE_NONE

MAMDA_BOOK_CHECK_TYPE_SNAPSHOT

MAMDA_BOOK_CHECK_TYPE_APPLY_DELTA

31
32 MAMDA_BOOK_CHECK_TYPE_NONE = 0,
33 MAMDA_BOOK_CHECK_TYPE_SNAPSHOT = 1,
34 MAMDA_BOOK_CHECK_TYPE_APPLY_DELTA = 2
35 };

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
5.1.3.12 enum Wombat::MamdaOptionAtTheMoneyCompareType

 Enumerator:

 \texttt{MAMDA\_AT\_THE\_MONEY\_COMPARE\_MID\_QUOTE}
 \texttt{MAMDA\_AT\_THE\_MONEY\_COMPARE\_BID}
 \texttt{MAMDA\_AT\_THE\_MONEY\_COMPARE\_ASK}
 \texttt{MAMDA\_AT\_THE\_MONEY\_COMPARE\_LAST\_TRADE}

\begin{verbatim}
29 {  
30  MAMDA_AT_THE_MONEY_COMPARE_MID_QUOTE = 0,
31  MAMDA_AT_THE_MONEY_COMPARE_BID = 1,
32  MAMDA_AT_THE_MONEY_COMPARE_ASK = 2,
33  MAMDA_AT_THE_MONEY_COMPARE_LAST_TRADE = 3
34  };
\end{verbatim}

5.1.3.13 enum Wombat::MamdaOptionPutCall

 Enumeration for indicating whether an option contract is a put or a call.

 Enumerator:

 \texttt{MAMDA\_PUT\_CALL\_CALL}
 \texttt{MAMDA\_PUT\_CALL\_PUT}
 \texttt{MAMDA\_PUT\_CALL\_UNKNOWN}

\begin{verbatim}
33 {  
34  MAMDA_PUT_CALL_CALL = 'C',
35  MAMDA_PUT_CALL_PUT = 'P',
36  MAMDA_PUT_CALL_UNKNOWN = 'Z'
37  };
\end{verbatim}

5.1.3.14 enum Wombat::MamdaOptionExerciseStyle

 Enumeration for indicating the style of an individual option contract.

 Enumerator:

 \texttt{MAMDA\_EXERCISE\_STYLE\_AMERICAN}
 \texttt{MAMDA\_EXERCISE\_STYLE\_EUROPEAN}
 \texttt{MAMDA\_EXERCISE\_STYLE\_CAPPED}
 \texttt{MAMDA\_EXERCISE\_STYLE\_UNKNOWN}
5.1.3.15 enum Wombat::MamdaNewsQueryType

Enumerator:

- QUERY_TYPE_UNKNOWN
- QUERY_TYPE_HISTORICAL
- QUERY_TYPE_SUBSCRIPTION
- QUERY_TYPE_HISTORICAL_SUBSCRIPTION

5.1.3.16 enum Wombat::MamdaNewsPriority

Enumerator:

- MAMDA_NEWS_PRIORITY_NONE
- MAMDA_NEWS_PRIORITY_NORMAL
- MAMDA_NEWS_PRIORITY_HOT

5.1.4 Function Documentation

5.1.4.1 MAMAExpDLL const char* Wombat::mamdaCheckTypeToString (MamdaCheckerType type)

5.1.4.2 MAMAExpDLL const char* Wombat::toString (MamdaFieldState fieldState)

Convert a MamdaFieldState to an appropriate, displayable string.
5.1 Wombat Namespace Reference

Parameters:

\[ \text{mamdaFieldState} \] The `MamdaFieldState` to stringify

Returns:

The stringified version of the `MamdaFieldState`

5.1.4.3 MAMDAExpDLL const char* Wombat::toString (MamdaSecurityStatus securityStatus)

Convert a `MamdaSecurityStatus` to an appropriate, displayable string.

Parameters:

\[ \text{securityStatus} \] The security status as an enumerated type.

5.1.4.4 MAMDAExpDLL MamdaSecurityStatus Wombat::mamdaSecurityStatusFromString (const char* securityStatusStr)

Convert a string representation of a security status to the enumeration.

This function is used internally for compatibility with older feed handler configurations, which may send the field as a string.

Parameters:

\[ \text{securityStatusStr} \] The security status as a string.

Returns:

The security status as an enumerated type.

5.1.4.5 MAMDAExpDLL const char* Wombat::toString (MamdaSecurityStatusQual securityStatusQual)

Convert a `MamdaSecurityStatusQual` to an appropriate, displayable string.

Parameters:

\[ \text{securityStatusQual} \] The security status qualifier as an enumeration.

Returns:

The security status qualifier as a string.
5.1.4.6 MAMDAExpDLL MamdaSecurityStatusQual
Wombat::mamdaSecurityStatusQualFromString (const char ∗securityStatusQualStr)

Convert a string representation of a security status to the enumeration.
This function is used internally for compatibility with older feed handler configurations, which may send the field as a string.

Parameters:

securityStatusQualStr The security status qualifier as a string.

Returns:

The security status qualifier as an enumeration.

5.1.4.7 MAMDAExpDLL const char ∗Wombat::toString
(MamdaTradeDirection tradeDir)

Convert a MamdaTradeDirection to an appropriate, displayable string.

Parameters:

tradeDir The MamdaTradeDirection to stringify

Returns:

The stringified version of the MamdaTradeDirection

5.1.4.8 MAMDAExpDLL MamdaTradeDirection
Wombat::mamdaTradeDirectionFromString (const char ∗tradeDirStr)

Convert a string representation of a trade direction to the enumeration.
This function is used internally for compatibility with older feed handler configurations, which may send the field as a string.

Parameters:

tradeDirStr The trade direction as a string.

Returns:

The trade direction as an enumerated value.
5.1.4.9 MAMDAExpDLL const char* Wombat::toString (MamdaTradeExecVenue tradeExecVenue)

Convert a MamdaTradeExecVenue to an appropriate, displayable string.

Parameters:

*tradeExecVenue* The MamdaTradeExecVenue to stringify

Returns:

The stringified version of the MamdaTradeExecVenue

5.1.4.10 MAMDAExpDLL MamdaTradeExecVenue Wombat::mamdaTradeExecVenueFromString (const char* tradeExecVenueStr)

Convert a string representation of a trade execution venue to the enumeration.

This function is used internally for compatibility with older feed handler configurations, which may send the field as a string.

Parameters:

*tradeExecVenueStr* The trade execution venue as a string.

Returns:

The trade execution venue as an enumerated value.

5.1.4.11 MAMDAExpDLL const char* Wombat::toString (MamdaTradeSide tradeSide)

Convert a MamdaTradeSide to an appropriate, displayable string.

Parameters:

*tradeSide* The MamdaTradeSide to stringify

Returns:

The stringified version of the MamdaTradeSide
### 5.1.4.12 MAMDAExpDLL MamdaTradeSide Wombat::mamdaTradeSideFromString (const char * tradeSideStr)

Convert a string representation of a trade side to the enumeration.

This function is used internally for compatibility with older feed handler configurations, which may send the field as a string.

**Parameters:**

- `tradeSideStr` The trade side as a string.

**Returns:**

The trade side as an enumerated value.

### 5.1.4.13 MAMDAExpDLL const char* Wombat::toString (MamdaUncrossPriceInd securityStatus)

Convert a MamdaUncrossPriceInd to an appropriate, displayable string.

**Parameters:**

- `securityStatus` The uncross price Ind as an enumerated type.

### 5.1.4.14 MAMDAExpDLL MamdaUncrossPriceInd Wombat::mamdaUncrossPriceIndFromString (const char * uncrossPriceInd)

Convert a string representation of an uncross price Ind to the enumeration.

This function is used internally for compatibility with older feed handler configurations, which may send the field as a string.

**Parameters:**

- `uncrossPriceInd` The uncross price Ind as a string.

**Returns:**

The uncross price Ind as an enumerated type.
5.1 Wombat Namespace Reference

5.1.4.15 MAMDAExpDLL const char* Wombat::getMamdaVersion (void)

Get the version of Mamda.

Returns:

- Version string as const char*

5.1.4.16 MAMAExpDLL const char* Wombat::mamdaOrderBookCheckTypeToString (MamdaOrderBookCheckType type)
Chapter 6

MAMDA C++ API Class Documentation

6.1  Wombat::char_str_less_than Struct Reference

#include <MamdaOptionContractSet.h>

Public Member Functions

- bool operator() (const char *str1, const char *str2) const

6.1.1  Member Function Documentation

6.1.1.1  bool Wombat::char_str_less_than::operator() (const char * str1, const char * str2) const

37  {
38       return strcmp(str1,str2) < 0;
39  }

The documentation for this struct was generated from the following file:

- MamdaOptionContractSet.h
6.2 Wombat::MamdaAndQuery Class Reference

#include <MamdaQuery.h>

Inheritance diagram for Wombat::MamdaAndQuery::

```
Wombat::MamdaQuery
    Wombat::MamdaAndQuery
```

Public Member Functions

- MamdaAndQuery (MamdaQuery *query1, MamdaQuery *query2)
- bool getXML (char *result)
- int getDepth ()
- void addQuery (MamdaQuery *query1)

6.2.1 Constructor & Destructor Documentation

6.2.1.1 Wombat::MamdaAndQuery::MamdaAndQuery (MamdaQuery *query1, MamdaQuery *query2)

6.2.2 Member Function Documentation

6.2.2.1 bool Wombat::MamdaAndQuery::getXML (char *result) [virtual]

Implements Wombat::MamdaQuery.

6.2.2.2 int Wombat::MamdaAndQuery::getDepth () [virtual]

Implements Wombat::MamdaQuery.

6.2.2.3 void Wombat::MamdaAndQuery::addQuery (MamdaQuery *query1)

The documentation for this class was generated from the following file:

- MamdaQuery.h
#include `<MamdaAuctionFields.h>`

**Static Public Member Functions**

- static void `setDictionary` (const MamaDictionary &`dictionary`)
- static void `reset` ()

`Reset the dictionary for fundamental update fields.`

- static uint16_t `getMaxFid` ()
- static bool `isSet` ()

**Static Public Attributes**

- static const MamaFieldDescriptor * `UNCROSS_PRICE`
- static const MamaFieldDescriptor * `UNCROSS_VOLUME`
- static const MamaFieldDescriptor * `UNCROSS_PRICE_IND`
- static const MamaFieldDescriptor * `AUCTION_TIME`

## 6.3.1 Member Function Documentation

### 6.3.1.1 static void Wombat::MamdaAuctionFields::setDictionary (const MamaDictionary & dictionary) [static]

### 6.3.1.2 static void Wombat::MamdaAuctionFields::reset () [static]

Reset the dictionary for fundamental update fields.
6.3.1.3 static uint16_t Wombat::MamdaAuctionFields::getMaxFid () [static]

6.3.1.4 static bool Wombat::MamdaAuctionFields::isSet () [static]

6.3.2 Member Data Documentation

6.3.2.1 const MamaFieldDescriptor* Wombat::MamdaAuctionFields::UNCROSS_PRICE [static]

6.3.2.2 const MamaFieldDescriptor* Wombat::MamdaAuctionFields::UNCROSS_VOLUME [static]

6.3.2.3 const MamaFieldDescriptor* Wombat::MamdaAuctionFields::UNCROSS_PRICE_IND [static]

6.3.2.4 const MamaFieldDescriptor* Wombat::MamdaAuctionFields::AUCTION_TIME [static]

The documentation for this class was generated from the following file:

- MamdaAuctionFields.h
MamdaAuctionHandler is an interface for applications that want to have an easy way to access currency data. 

#include <MamdaAuctionHandler.h>

Public Member Functions

• virtual void onAuctionRecap (MamdaSubscription *subscription, MamdaAuctionListener &listener, const MamaMsg &msg, const MamdaAuctionRecap &recap)=0

  Method invoked when the current auction information for the security is available.

• virtual void onAuctionUpdate (MamdaSubscription *subscription, MamdaAuctionListener &listener, const MamaMsg &msg, const MamdaAuctionRecap &recap, const MamdaAuctionUpdate &update)=0

  Method invoked when one or more of the Auction fields have been updated by one of the following market data events:

  – Initial image.

• virtual ~MamdaAuctionHandler ()

6.4.1 Detailed Description

MamdaAuctionHandler is an interface for applications that want to have an easy way to access currency data.

The interface defines callback methods for for receiving updates on currency data.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 virtual Wombat::MamdaAuctionHandler::~MamdaAuctionHandler ()

[virtual]

87 {};

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.4.3 Member Function Documentation

6.4.3.1 virtual void Wombat::MamdaAuctionHandler::onAuctionRecap
(MamdaSubscription * subscription, MamdaAuctionListener & listener,
const MamaMsg & msg, const MamdaAuctionRecap & recap) [pure virtual]

Method invoked when the current auction information for the security is available.

The reason for the invocation may be any of the following:

- Initial image.
- Recap update (e.g., after server fault tolerant event or data quality event.)
- After stale status removed.

Parameters:

subscription The subscription which received the update.
listener The listener which invoked this callback.
msg The MamaMsg that triggered this invocation.
recap Access to the full auction recap details.

6.4.3.2 virtual void Wombat::MamdaAuctionHandler::onAuctionUpdate
(MamdaSubscription * subscription, MamdaAuctionListener & listener,
const MamaMsg & msg, const MamdaAuctionRecap & recap, const
MamdaAuctionUpdate & update) [pure virtual]

Method invoked when one or more of the Auction fields have been updated by one of
the following market data events:

- Initial image.
- Recap update (e.g., after server fault tolerant event or data quality event.)
- Generic update.

Parameters:

subscription The subscription which received the update.
listener The listener which invoked this callback.
msg The MamaMsg that triggered this invocation.
recap Access to the full auction recap details.
**update**  Access to the auction update details.

The documentation for this class was generated from the following file:

- MamdaAuctionHandler.h
6.5 Wombat::MamdaAuctionListener Class Reference

MamdaAuctionListener is a class that specializes in handling currency data, Developers provide their own implementation of the MamdaAuctionHandler interface and will be delivered notifications for updates in the currency data.

```cpp
#include <MamdaAuctionListener.h>
```

Inheritance diagram for Wombat::MamdaAuctionListener::

```
Wombat::MamdaBasicRecap  Wombat::MamdaBasicEvent
Wombat::MamdaMsgListener  Wombat::MamdaAuctionRecap  Wombat::MamdaAuctionUpdate
Wombat::MamdaAuctionListener
```

### Public Member Functions

- `MamdaAuctionListener ()`
- virtual `~MamdaAuctionListener ()`
- void `addHandler (MamdaAuctionHandler *handler)`
- `const char * getSymbol () const`
    
    Get the instruments string symbol.

- `const char * getPartId () const`
    
    Get the participant identifier.

- `const MamaDateTime & getSrcTime () const`
    
    Get the source time.

- `const MamaDateTime & getActivityTime () const`
    
    Get the activity time.

- `const MamaDateTime & getLineTime () const`
    
    Get the line time.

- `const MamaDateTime & getSendTime () const`
    
    Get the send time.

- `mama_seqnum_t getEventSeqNum () const`
    
    Get the event sequence number.
- `const MamaDateTime & getEventTime () const`  
  Get the event time.

- `const MamaMsgQual & getMsgQual () const`  
  Get the message qualifier.

- `bool isInitialised () const`  
  `MamdaFieldState getSymbolFieldState () const`  
  Get the string symbol field state.

- `MamdaFieldState getPartIdFieldState () const`  
  Get the participant identifier field state.

- `MamdaFieldState getSrcTimeFieldState () const`  
  Get the source time field state.

- `MamdaFieldState getActivityTimeFieldState () const`  
  Get the activity time field state.

- `MamdaFieldState getLineTimeFieldState () const`  
  Get the line time of the update.

- `MamdaFieldState getSendTimeFieldState () const`  
  Get the send time field state.

- `MamdaFieldState getEventSeqNumFieldState () const`  
  Get the event sequence number field state.

- `MamdaFieldState getEventTimeFieldState () const`  
  Get the event time field state.

- `MamdaFieldState getMsgQualFieldState () const`  
  Get the message qualifier field state.

- `const MamaPrice & getUncrossPrice () const`  
  Get the uncross price.

- `mama_quantity_t getUncrossVolume () const`  
  Get the uncross volume.

- `MamdaUncrossPriceInd getUncrossPriceInd () const`
Get the uncross price Ind.

- const MamaDateTime & getAuctionTime () const
  
  Get the auction time.

- MamdaFieldState getUncrossPriceFieldState () const
  
  Get the uncross price fieldState.

- MamdaFieldState getUncrossVolumeFieldState () const
  
  Get the uncross volume fieldState.

- MamdaFieldState getUncrossPriceIndFieldState () const
  
  Get the uncross price ind fieldState.

- MamdaFieldState getAuctionTimeFieldState () const
  
  Get the auction time fieldState.

- virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)
  
  Implementation of MamdaListener interface.

### 6.5.1 Detailed Description

MamdaAuctionListener is a class that specializes in handling currency data. Developers provide their own implementation of the MamdaAuctionHandler interface and will be delivered notifications for updates in the currency data.

An obvious application for this MAMDA class is any kind of currency analysis application.

MamdaAuctionListener should initialize the MamdaAuctionFields class prior to receiving the first message by calling MamdaAuctionFields::setDictionary() with a valid dictionary object which contains Auction related fields.
6.5 Wombat::MamdaAuctionListener Class Reference

6.5.2 Constructor & Destructor Documentation

6.5.2.1 Wombat::MamdaAuctionListener::MamdaAuctionListener ()

6.5.2.2 virtual Wombat::MamdaAuctionListener::~MamdaAuctionListener () [virtual]

6.5.3 Member Function Documentation

6.5.3.1 void Wombat::MamdaAuctionListener::addHandler (MamdaAuctionHandler * handler)

6.5.3.2 const char* Wombat::MamdaAuctionListener::getSymbol () const [virtual]

Get the instruments string symbol.

Returns:

Symbol. This is the "well-known" symbol for the security, including any symbolology mapping performed by the publisher.

Implements Wombat::MamdaBasicEvent.

6.5.3.3 const char* Wombat::MamdaAuctionListener::getPartId () const [virtual]

Get the participant identifier.

Returns:

Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).

Implements Wombat::MamdaBasicEvent.

6.5.3.4 const MamaDateTime& Wombat::MamdaAuctionListener::getSrcTime () const [virtual]

Get the source time.

Returns:

Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.
Implements Wombat::MamdaBasicEvent.

6.5.3.5 const MamaDateTime& Wombat::MamdaAuctionListener::getActivityTime() const [virtual]

Get the activity time.

Returns:
Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements Wombat::MamdaBasicEvent.

6.5.3.6 const MamaDateTime& Wombat::MamdaAuctionListener::getLineTime() const [virtual]

Get the line time.

Returns:
Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

6.5.3.7 const MamaDateTime& Wombat::MamdaAuctionListener::getSendTime() const [virtual]

Get the send time.

Returns:
Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime()).

Implements Wombat::MamdaBasicEvent.
Get the event sequence number.

**Returns:**

Source sequence number. The exchange generated sequence number.

Implements Wombat::MamdaBasicEvent.

Get the event time.

**Returns:**

Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

Get the message qualifier.

**Returns:**

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

Get the string symbol field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.
Implements Wombat::MamdaBasicEvent.

6.5.3.13 MamdaFieldState Wombat::MamdaAuctionListener::getPartIdFieldState () const [virtual]

Get the participant identifier field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.5.3.14 MamdaFieldState Wombat::MamdaAuctionListener::getSrcTimeFieldState () const [virtual]

Get the source time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.5.3.15 MamdaFieldState Wombat::MamdaAuctionListener::getActivityTimeFieldState () const [virtual]

Get the activity time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.5.3.16 MamdaFieldState Wombat::MamdaAuctionListener::getLineTimeFieldState () const [virtual]

Get the line time of the update.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.5 Wombat::MamdaAuctionListener Class Reference

6.5.3.17 **MamdaFieldState** Wombat::MamdaAuctionListener::getSendTimeFieldState () const [virtual]

Get the send time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaBasicEvent**.

6.5.3.18 **MamdaFieldState** Wombat::MamdaAuctionListener::getEventSeqNumFieldState () const [virtual]

Get the event sequence number field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaBasicEvent**.

6.5.3.19 **MamdaFieldState** Wombat::MamdaAuctionListener::getEventTimeFieldState () const [virtual]

Get the event time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaBasicEvent**.

6.5.3.20 **MamdaFieldState** Wombat::MamdaAuctionListener::getMsgQualFieldState () const [virtual]

Get the message qualifier field state.

**Returns:**

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements **Wombat::MamdaBasicEvent**.
6.5.3.21 const MamaPrice& Wombat::MamdaAuctionListener::getUncrossPrice () const [virtual]

Get the uncross price.

Returns:
uncross price. The indicative or firm auction price.

Implements Wombat::MamdaAuctionRecap.

6.5.3.22 mama_quantity_t Wombat::MamdaAuctionListener::getUncrossVolume () const [virtual]

Get the uncross volume.

Returns:
Ask price. The indicative volume, or the volume turned over in the auction

Implements Wombat::MamdaAuctionRecap.

6.5.3.23 MamaUncrossPriceInd Wombat::MamdaAuctionListener::getUncrossPriceInd () const [virtual]

Get the uncross price Ind.

Returns:
uncross price Ind. Indicates whether the Price and Volume is an indicative of the current state of the auction or whether its the (firm) auction price and volume

Implements Wombat::MamdaAuctionRecap.

6.5.3.24 const MamaDateTime& Wombat::MamdaAuctionListener::getAuctionTime () const [virtual]

Get the auction time.

Returns:
auction time. The indicative or firm auction time.

Implements Wombat::MamdaAuctionRecap.
6.5.3.25 MamdaFieldState Wombat::MamdaAuctionListener::getUncrossPriceFieldState () const [virtual]

Get the uncross price fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaAuctionRecap.

6.5.3.26 MamdaFieldState Wombat::MamdaAuctionListener::getUncrossVolumeFieldState () const [virtual]

Get the uncross volume fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaAuctionRecap.

6.5.3.27 MamdaFieldState Wombat::MamdaAuctionListener::getUncrossPriceIndFieldState () const [virtual]

Get the uncross price ind fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaAuctionRecap.

6.5.3.28 MamdaFieldState Wombat::MamdaAuctionListener::getAuctionTimeFieldState () const [virtual]

Get the auction time fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaAuctionRecap.
6.5.3.29 virtual void Wombat::MamdaAuctionListener::onMsg
(MamdaSubscription * subscription, const MamaMsg & msg, short
msgType) [virtual]

Implementation of MamdaListener interface.

Exceptions:

<.MamaStatus> {Wombat default implementation.}

Implements Wombat::MamdaMsgListener.

The documentation for this class was generated from the following file:

* MamdaAuctionListener.h
MamdaAuctionRecap is an interface that provides access to the currency related fields.

#include <MamdaAuctionRecap.h>

Inheritance diagram for Wombat::MamdaAuctionRecap:

```
Wombat::MamdaBasicRecap

Wombat::MamdaAuctionRecap

Wombat::MamdaAuctionListener
```

Public Member Functions

- virtual const MamaPrice & getUncrossPrice() const =0
  
  Get the uncross price.

- virtual mama_quantity_t getUncrossVolume() const =0
  
  Get the uncross volume.

- virtual MamdaUncrossPriceInd getUncrossPriceInd() const =0
  
  Get the uncross price Ind.

- virtual const MamaDateTime & getAuctionTime() const =0
  
  Get the auction time.

- virtual MamdaFieldState getUncrossPriceFieldState() const =0
  
  Get the uncross price fieldState.

- virtual MamdaFieldState getUncrossVolumeFieldState() const =0
  
  Get the uncross volume fieldState.

- virtual MamdaFieldState getUncrossPriceIndFieldState() const =0
  
  Get the uncross price ind fieldState.

- virtual MamdaFieldState getAuctionTimeFieldState() const =0
  
  Get the auction time fieldState.

- virtual ~MamdaAuctionRecap()
6.6.1 Detailed Description

*MamdaAuctionRecap* is an interface that provides access to the currency related fields.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 virtual Wombat::MamdaAuctionRecap::~MamdaAuctionRecap ()
[virtual]

98 {};

6.6.3 Member Function Documentation

6.6.3.1 virtual const MamaPrice& Wombat::MamdaAuctionRecap::getUncrossPrice () const [pure virtual]

Get the uncross price.

**Returns:**

uncross price. The indicative or firm auction price.

Implemented in *Wombat::MamdaAuctionListener*.

6.6.3.2 virtual mama_quantity_t Wombat::MamdaAuctionRecap::getUncrossVolume () const [pure virtual]

Get the uncross volume.

**Returns:**

Ask price. The indicative volume, or the volume turned over in the auction

Implemented in *Wombat::MamdaAuctionListener*.

6.6.3.3 virtual MamdaUncrossPriceInd Wombat::MamdaAuctionRecap::getUncrossPriceInd () const [pure virtual]

Get the uncross price Ind.
Returns:

uncross price Ind. Indicates whether the Price and Volume is an indicative of the current state of the auction or whether its the (firm) auction price and volume.

Implemented in Wombat::MamdaAuctionListener.

6.6.3.4 virtual const MamaDateTime& Wombat::MamdaAuctionRecap::getAuctionTime () const [pure virtual]

Get the auction time.

Returns:

auction time. The indicative or firm auction time.

Implemented in Wombat::MamdaAuctionListener.

6.6.3.5 virtual MamdaFieldState Wombat::MamdaAuctionRecap::getUncrossPriceFieldState () const [pure virtual]

Get the uncross price fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener.

6.6.3.6 virtual MamdaFieldState Wombat::MamdaAuctionRecap::getUncrossVolumeFieldState () const [pure virtual]

Get the uncross volume fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener.
6.6.3.7 virtual MamdaFieldState Wombat::MamdaAuctionRecap::getUncrossPriceIndFieldState () const [pure virtual]

Get the uncross price ind fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener.

6.6.3.8 virtual MamdaFieldState Wombat::MamdaAuctionRecap::getAuctionTimeFieldState () const [pure virtual]

Get the auction time fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener.

The documentation for this class was generated from the following file:

- MamdaAuctionRecap.h
6.7  Wombat::MamdaAuctionUpdate Class Reference

MamdaAuctionUpdate is an interface that provides access to the currency related fields.

#include <MamdaAuctionUpdate.h>

Inheritance diagram for Wombat::MamdaAuctionUpdate:

\[\text{Wombat::MamdaBasicEvent} \rightarrow \text{Wombat::MamdaAuctionUpdate} \rightarrow \text{Wombat::MamdaAuctionListener}\]

Public Member Functions

- virtual const MamaPrice & \text{getUncrossPrice} () const =0
  
  \text{Get the uncross price.}

- virtual mama_quantity_t \text{getUncrossVolume} () const =0
  
  \text{Get the uncross volume.}

- virtual MamdaUncrossPriceInd \text{getUncrossPriceInd} () const =0
  
  \text{Get the uncross price Ind.}

- virtual const MamaDateTime & \text{getAuctionTime} () const =0
  
  \text{Get the auction time.}

- virtual MamdaFieldState \text{getUncrossPriceFieldState} () const =0
  
  \text{Get the uncross price fieldState.}

- virtual MamdaFieldState \text{getUncrossVolumeFieldState} () const =0
  
  \text{Get the uncross vol fieldState.}

- virtual MamdaFieldState \text{getUncrossPriceIndFieldState} () const =0
  
  \text{Get the uncross price ind fieldState.}

- virtual MamdaFieldState \text{getAuctionTimeFieldState} () const =0
  
  \text{Get the auction time fieldState.}

- virtual \text{~MamdaAuctionUpdate} ()
6.7.1 Detailed Description

MamdaAuctionUpdate is an interface that provides access to the currency related fields.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 virtual Wombat::MamdaAuctionUpdate::~MamdaAuctionUpdate () [virtual]

99 {};

6.7.3 Member Function Documentation

6.7.3.1 virtual const MamaPrice& Wombat::MamdaAuctionUpdate::getUncrossPrice () const [pure virtual]

Get the uncross price.

Returns:

uncross price. The indicative or firm auction price.

Implemented in Wombat::MamdaAuctionListener.

6.7.3.2 virtual mama_quantity_t Wombat::MamdaAuctionUpdate::getUncrossVolume () const [pure virtual]

Get the uncross volume.

Returns:

Ask price. The indicative volume, or the volume turned over in the auction

Implemented in Wombat::MamdaAuctionListener.

6.7.3.3 virtual MamdaUncrossPriceInd Wombat::MamdaAuctionUpdate::getUncrossPriceInd () const [pure virtual]

Get the uncross price Ind.
Returns:

uncross price Ind. Indicates whether the Price and Volume is an indicative of the current state of the auction or whether its the (firm) auction price and volume

Implemented in Wombat::MamdaAuctionListener.

6.7.3.4 virtual const MamaDateTime& Wombat::MamdaAuctionUpdate::getAuctionTime () const [pure virtual]

Get the auction time.

Returns:

auction time. The indicative or firm auction time .

Implemented in Wombat::MamdaAuctionListener.

6.7.3.5 virtual MamdaFieldState Wombat::MamdaAuctionUpdate::getUncrossPriceFieldState () const [pure virtual]

Get the uncross price fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener.

6.7.3.6 virtual MamdaFieldState Wombat::MamdaAuctionUpdate::getUncrossVolumeFieldState () const [pure virtual]

Get the uncross vol fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener.
6.7.3.7 virtual MamdaFieldState Wombat::MamdaAuctionUpdate::getUncrossPriceIndFieldState () const [pure virtual]

Get the uncross price ind fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener.

6.7.3.8 virtual MamdaFieldState Wombat::MamdaAuctionUpdate::getAuctionTimeFieldState () const [pure virtual]

Get the auction time fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener.

The documentation for this class was generated from the following file:

- MamdaAuctionUpdate.h
6.8 Wombat::MamdaBasicErrorListener Class Reference

MamdaBasicErrorListener defines an interface for handling error notifications for a MamdaBasicSubscription.

#include <MamdaErrorListener.h>

Public Member Functions

• virtual void onError (MamdaBasicSubscription *subscription, MamdaErrorSeverity severity, MamdaErrorCode errorCode, const char *errorStr)=0
  
  Provide a callback to handle errors.

• virtual ~MamdaBasicErrorListener ()

6.8.1 Detailed Description

MamdaBasicErrorListener defines an interface for handling error notifications for a MamdaBasicSubscription.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 virtual Wombat::MamdaBasicErrorListener::~MamdaBasicErrorListener () [virtual]

99 {};

6.8.3 Member Function Documentation

6.8.3.1 virtual void Wombat::MamdaBasicErrorListener::onError
  (MamdaBasicSubscription *subscription, MamdaErrorSeverity severity, MamdaErrorCode errorCode, const char *errorStr) [pure virtual]

Provide a callback to handle errors.

The severity is intended as a hint to indicate whether the error is recoverable.

The documentation for this class was generated from the following file:

• MamdaErrorListener.h
6.9 Wombat::MamdaBasicEvent Class Reference

MamdaBasicEvent is a superclass interface that provides access to common event related fields.

#include <MamdaBasicEvent.h>

Inheritance diagram for Wombat::MamdaBasicEvent:

Public Member Functions

• virtual const char * getSymbol () const =0

Get the instruments string symbol.

• virtual MamdaFieldState getSymbolFieldState () const =0
Get the string symbol field state.

- virtual const char * getPartId () const =0
  Get the participant identifier.

- virtual MamdaFieldState getPartIdFieldState () const =0
  Get the participant identifier field state.

- virtual mama_seqnum_t getEventSeqNum () const =0
  Get the event sequence number.

- virtual MamdaFieldState getEventSeqNumFieldState () const =0
  Get the event sequence number field state.

- virtual const MamaDateTime & getEventTime () const =0
  Get the event time.

- virtual MamdaFieldState getEventTimeFieldState () const =0
  Get the event time field state.

- virtual const MamaDateTime & getSrcTime () const =0
  Get the source time.

- virtual MamdaFieldState getSrcTimeFieldState () const =0
  Get the source time field state.

- virtual const MamaDateTime & getActivityTime () const =0
  Get the activity time.

- virtual MamdaFieldState getActivityTimeFieldState () const =0
  Get the activity time field state.

- virtual const MamaDateTime & getLineTime () const =0
  Get the line time.

- virtual MamdaFieldState getLineTimeFieldState () const =0
  Get the line time of the update.

- virtual const MamaDateTime & getSendTime () const =0
  Get the send time.

- virtual MamdaFieldState getSendTimeFieldState () const =0
Get the send time field state.

- virtual const MamaMsgQual & getMsgQual () const =0
  Get the message qualifier.

- virtual MamdaFieldState getMsgQualFieldState () const =0
  Get the message qualifier field state.

- virtual ~MamdaBasicEvent ()
  Destructor.

6.9.1 Detailed Description

MamdaBasicEvent is a superclass interface that provides access to common event related fields.

Events types include trades, quotes, order book updates, closing summaries, etc. See the individual subclasses for specific information about each event type.

Note: Different types of time stamps are available representing the time that the event (trade, quote, etc) actually occurred, the time the data source/exchange reported it, and the time the NYSE Technologies feed handler applied an action to the given data item (record, order book, etc.). Many feeds to not provide a distinction between the event time and the source time (they may be the same) and the granularity of time stamps also varies between data sources.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 virtual Wombat::MamdaBasicEvent::~MamdaBasicEvent ()
[virtual]

Destructor.

205 {

6.9.3 Member Function Documentation

6.9.3.1 virtual const char* Wombat::MamdaBasicEvent::getSymbol () const
[pure virtual]

Get the instruments string symbol.
Returns:

Symbol. This is the "well-known" symbol for the security, including any symbolism mapping performed by the publisher.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.2 virtual MamdaFieldState Wombat::MamdaBasicEvent::getSymbolFieldState () const [pure virtual]

Get the string symbol field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.3 virtual const char∗ Wombat::MamdaBasicEvent::getPartId () const [pure virtual]

Get the participant identifier.

Returns:

Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaOrderImbalanceUpdate, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener,
Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.4 virtual MamdaFieldState Wombat::MamdaBasicEvent::getPartIdFieldState () const [pure virtual]

Get the participant identifier field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaOrderImbalanceUpdate, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.5 virtual mama_seqnum_t Wombat::MamdaBasicEvent::getEventSeqNum () const [pure virtual]

Get the event sequence number.

Returns:
Source sequence number. The exchange generated sequence number.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaOrderImbalanceUpdate, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.6 virtual MamdaFieldState Wombat::MamdaBasicEvent::getEventSeqNumFieldState () const [pure virtual]

Get the event sequence number field state.
Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaOrderImbalanceUpdate, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.7 virtual const MamaDateTime& Wombat::MamdaBasicEvent::getEventTime () const [pure virtual]

Get the event time.

Returns:

Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaOrderImbalanceUpdate, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.8 virtual MamdaFieldState Wombat::MamdaBasicEvent::getEventTimeFieldState () const [pure virtual]

Get the event time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaOrderImbalanceUpdate, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.
Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.9 virtual const MamaDateTime& Wombat::MamdaBasicEvent::getSrcTime () const [pure virtual]

Get the source time.

Returns:
Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.10 virtual MamdaFieldState Wombat::MamdaBasicEvent::getSrcTimeFieldState () const [pure virtual]

Get the source time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.11 virtual const MamaDateTime& Wombat::MamdaBasicEvent::getActivityTime () const [pure virtual]

Get the activity time.
6.9 Wombat::MamdaBasicEvent Class Reference

Returns:

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.12 virtual MamdaFieldState Wombat::MamdaBasicEvent::getActivityTimeFieldState () const [pure virtual]

Get the activity time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.13 virtual const MamaDateTime& Wombat::MamdaBasicEvent::getLineTime () const [pure virtual]

Get the line time.

Returns:

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrder...
**6.9.3.14 virtual MamdaFieldState Wombat::MamdaBasicEvent::getLineTimeFieldState () const [pure virtual]**

Get the line time of the update.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

**6.9.3.15 virtual const MamaDateTime& Wombat::MamdaBasicEvent::getSendTime () const [pure virtual]**

Get the send time.

**Returns:**

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime().

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.
6.9 Wombat::MamdaBasicEvent Class Reference

6.9.3.16 virtual MamdaFieldState Wombat::MamdaBasicEvent::getSendTimeFieldState () const [pure virtual]

Get the send time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.17 virtual const MamaMsgQual& Wombat::MamdaBasicEvent::getMsgQual () const [pure virtual]

Get the message qualifier.

Returns:
Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaConcreteBasicEvent, Wombat::MamdaCurrencyListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, Wombat::MamdaOrderBookConcreteComplexDelta, Wombat::MamdaOrderBookConcreteSimpleDelta, Wombat::MamdaOptionChainListener, and Wombat::MamdaNewsHeadline.

6.9.3.18 virtual MamdaFieldState Wombat::MamdaBasicEvent::getMsgQualFieldState () const [pure virtual]

Get the message qualifier field state.

Returns:
Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.
Implemented in `Wombat::MamdaAuctionListener`, `Wombat::MamdaConcreteBasicEvent`, `Wombat::MamdaCurrencyListener`, `Wombat::MamdaOrderImbalanceListener`, `Wombat::MamdaQuoteListener`, `Wombat::MamdaSecStatusListener`, `Wombat::MamdaTradeListener`, `Wombat::MamdaBookAtomicListener`, `Wombat::MamdaOrderBookConcreteComplexDelta`, `Wombat::MamdaOrderBookConcreteSimpleDelta`, `Wombat::MamdaOptionChainListener`, and `Wombat::MamdaNewsHeadline`.

The documentation for this class was generated from the following file:

- `MamdaBasicEvent.h`
MamdaBasicMsgListener defines an interface for handling MAMA messages for a MamdaBasicSubscription.

#include <MamdaMsgListener.h>

Public Member Functions

- virtual void onMsg (MamdaBasicSubscription *subscription, const MamaMsg &msg, short msgType)=0
- virtual ~MamdaBasicMsgListener ()

6.10.1 Detailed Description

MamdaBasicMsgListener defines an interface for handling MAMA messages for a MamdaBasicSubscription.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 virtual Wombat::MamdaBasicMsgListener::~MamdaBasicMsgListener () [virtual]

60 {};

6.10.3 Member Function Documentation

6.10.3.1 virtual void Wombat::MamdaBasicMsgListener::onMsg (MamdaBasicSubscription * subscription, const MamaMsg & msg, short msgType) [pure virtual]

The documentation for this class was generated from the following file:

- MamdaMsgListener.h
6.11 Wombat::MamdaBasicQualityListener Class Reference

MamdaBasicQualityListener defines an interface for handling changes in quality notifications for a MamdaBasicSubscription.

#include <MamdaQualityListener.h>

Public Member Functions

- virtual void onQuality (MamdaBasicSubscription *subscription, mamaQuality quality)=0
- virtual ~MamdaBasicQualityListener ()

6.11.1 Detailed Description

MamdaBasicQualityListener defines an interface for handling changes in quality notifications for a MamdaBasicSubscription.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 virtual Wombat::MamdaBasicQualityListener::~MamdaBasicQualityListener () [virtual]

58 {};

6.11.3 Member Function Documentation

6.11.3.1 virtual void Wombat::MamdaBasicQualityListener::onQuality (MamdaBasicSubscription * subscription, mamaQuality quality) [pure virtual]

The documentation for this class was generated from the following file:

- MamdaQualityListener.h
MamdaBasicRecap is an interface that provides access to recap related fields.

#include <MamdaBasicRecap.h>

Inheritance diagram for Wombat::MamdaBasicRecap::

Public Member Functions

- virtual const char * getSymbol () const =0
  Get the string symbol for the instrument.

- virtual MamdaFieldState getSymbolFieldState () const =0
  Get the string symbol field state for the instrument.

- virtual const char * getPartId () const =0
  Get the participant identifier.
• virtual MamdaFieldState getPartIdFieldState () const =0
  Get the participant identifier field state.

• virtual const MamaDateTime & getSrcTime () const =0
  Get the source time of the update.

• virtual MamdaFieldState getSrcTimeFieldState () const =0
  Get the source time field state.

• virtual const MamaDateTime & getActivityTime () const =0
  Get the activity time of the update.

• virtual MamdaFieldState getActivityTimeFieldState () const =0
  Get the activity time field state.

• virtual const MamaDateTime & getLineTime () const =0
  Get the line time of the update.

• virtual MamdaFieldState getLineTimeFieldState () const =0
  Get the line time field state.

• virtual const MamaDateTime & getSendTime () const =0
  Get the send time of the update.

• virtual MamdaFieldState getSendTimeFieldState () const =0
  Get the send time field state.

• virtual ~MamdaBasicRecap ()
  Destructor.

### 6.12.1 Detailed Description

MamdaBasicRecap is an interface that provides access to recap related fields.

### 6.12.2 Constructor & Destructor Documentation

#### 6.12.2.1 virtual Wombat::MamdaBasicRecap::~MamdaBasicRecap ()

[virtual]

Destructor.
6.12.3 Member Function Documentation

6.12.3.1 virtual const char∗ Wombat::MamdaBasicRecap::getSymbol () const
[pure virtual]

Get the string symbol for the instrument.

Returns:
Symbol. This is the "well-known" symbol for the security, including any symbol-
ology mapping performed by the publisher.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrency-
Listener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalance-
Listener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener,
Wombat::MamdaTradeListener, and Wombat::MamdaBookAtomicListener.

6.12.3.2 virtual MamdaFieldState Wombat::MamdaBasicRecap::getSymbolFieldState () const
[pure virtual]

Get the string symbol field state for the instrument.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrency-
Listener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalance-
Listener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener,
Wombat::MamdaTradeListener, and Wombat::MamdaBookAtomicListener.

6.12.3.3 virtual const char∗ Wombat::MamdaBasicRecap::getPartId () const
[pure virtual]

Get the participant identifier.

Returns:
Participant ID. This may be an exchange identifier, a market maker ID, etc., or
NULL (if this is not related to any specific participant).

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrency-
Listener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalance-
Listener, Wombat::MamdaOrderImbalanceRecap, Wombat::MamdaQuote-
Listener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener,
and Wombat::MamdaBookAtomicListener.
6.12.3.4 virtual MamdaFieldState Wombat::MamdaBasicRecap::getPartIdFieldState () const [pure virtual]

Get the participant identifier field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrencyListener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaOrderImbalanceRecap, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, and Wombat::MamdaBookAtomicListener.

6.12.3.5 virtual const MamaDateTime& Wombat::MamdaBasicRecap::getSrcTime () const [pure virtual]

Get the source time of the update.

Returns:

Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrencyListener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaOrderImbalanceRecap, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, and Wombat::MamdaBookAtomicListener.

6.12.3.6 virtual MamdaFieldState Wombat::MamdaBasicRecap::getSrcTimeFieldState () const [pure virtual]

Get the source time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrencyListener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaOrderImbalanceRecap, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, and Wombat::MamdaBookAtomicListener.
6.12.3.7 virtual const MamaDateTime& Wombat::MamdaBasicRecap::getActivityTime () const [pure virtual]

Get the activity time of the update.

**Returns:**

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrencyListener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, and Wombat::MamdaBookAtomicListener.

6.12.3.8 virtual MamdaFieldState Wombat::MamdaBasicRecap::getActivityTimeFieldState () const [pure virtual]

Get the activity time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrencyListener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, and Wombat::MamdaBookAtomicListener.

6.12.3.9 virtual const MamaDateTime& Wombat::MamdaBasicRecap::getLineTime () const [pure virtual]

Get the line time of the update.

**Returns:**

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.
Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrencyListener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, and Wombat::MamdaBook AtomicListener.

6.12.3.10 virtual MamdaFieldState Wombat::MamdaBasicRecap::getLineTimeFieldState () const [pure virtual]

Get the line time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrencyListener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, and Wombat::MamdaBook AtomicListener.

6.12.3.11 virtual const MamaDateTime& Wombat::MamdaBasicRecap::getSendTime () const [pure virtual]

Get the send time of the update.

Returns:

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime().

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrencyListener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, and Wombat::MamdaBook AtomicListener.

6.12.3.12 virtual MamdaFieldState Wombat::MamdaBasicRecap::getSendTimeFieldState () const [pure virtual]

Get the send time field state.
Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrencyListener, Wombat::MamdaFundamentalListener, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, and Wombat::MamdaBookAtomicListener.

The documentation for this class was generated from the following file:

- MamdaBasicRecap.h
6.13 Wombat::MamdaBasicSubscription Class Reference

A MamdaBasicSubscription is used to register interest in a particular symbol.

#include <MamdaBasicSubscription.h>

Public Member Functions

• MamdaBasicSubscription ()
• ~MamdaBasicSubscription ()
• void create (MamaTransport *transport, MamaQueue *queue, const char *symbol, void *closure=NULL)
  
  Create and activate the subscription.

• void setTimeout (double timeout)
  
  Set the subscription timeout (in seconds).

• void addMsgListener (MamdaBasicMsgListener *listener)
  
  Add a listener for regular messages.

• void addQualityListener (MamdaBasicQualityListener *listener)
  
  Add a listener for changes in quality status.

• void addErrorListener (MamdaBasicErrorListener *listener)
  
  Add a listener for error events.

• void activate ()
  
  Activate the subscription.

• void deactivate ()
  
  Deactivate the subscription.

• void setSymbol (const char *symbol)
  
  Set the symbol for the subscription.

• const char * getSymbol () const
  
  Get the symbol for the subscription.

• void setClosure (void *closure)
  
  Set the closure for the subscription.
6.13 Wombat::MamdaBasicSubscription Class Reference

- void * getClosure () const
  
  Get the additional object passed as the closure to the `create()` method.

- void setTransport (MamaTransport *transport)
  
  Set the MAMA transport.

- MamaTransport * getTransport () const
  
  Get the MamaTransport for this subscription.

- void setQueue (MamaQueue *queue)
  
  Set the MAMA queue.

- MamaQueue * getQueue () const
  
  Get the MAMA queue for this subscription.

6.13.1 Detailed Description

A `MamdaBasicSubscription` is used to register interest in a particular symbol. A MamaTransport is required to actually activate the subscription. Multiple listeners can be added to the `MamdaBasicSubscription`. In this way, an application can make use of more than one of the specialized value added MAMDA listeners, such as MamdaCalcListener.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 Wombat::MamdaBasicSubscription::MamdaBasicSubscription ()

6.13.2.2 Wombat::MamdaBasicSubscription::~MamdaBasicSubscription ()

6.13.3 Member Function Documentation

6.13.3.1 void Wombat::MamdaBasicSubscription::create (MamaTransport * transport, MamaQueue * queue, const char * symbol, void * closure = NULL)

Create and activate the subscription.

You can alternatively set all objects and properties and then call activate is you wish.
6.13.3.2  void Wombat::MamdaBasicSubscription::setTimeout (double timeout)

Set the subscription timeout (in seconds).
Do this before calling activate().

6.13.3.3  void Wombat::MamdaBasicSubscription::addMsgListener
(MamdaBasicMsgListener * listener)

Add a listener for regular messages.

6.13.3.4  void Wombat::MamdaBasicSubscription::addQualityListener
(MamdaBasicQualityListener * listener)

Add a listener for changes in quality status.

6.13.3.5  void Wombat::MamdaBasicSubscription::addErrorListener
(MamdaBasicErrorListener * listener)

Add a listener for error events.

6.13.3.6  void Wombat::MamdaBasicSubscription::activate ()

Activate the subscription.
Until this method is invoked, no updates will be received. Is called by create().

6.13.3.7  void Wombat::MamdaBasicSubscription::deactivate ()

Deactivate the subscription.
No more updates will be received for this subscription (unless activate() is invoked again).

6.13.3.8  void Wombat::MamdaBasicSubscription::setSymbol (const char * symbol)

Set the symbol for the subscription.
Do this before calling activate().
6.13 Wombat::MamdaBasicSubscription Class Reference

6.13.3.9 const char* Wombat::MamdaBasicSubscription::getSymbol () const
Get the symbol for the subscription.

6.13.3.10 void Wombat::MamdaBasicSubscription::setClosure (void * closure)
Set the closure for the subscription.
Do this before calling activate().

6.13.3.11 void* Wombat::MamdaBasicSubscription::getClosure () const
Get the additional object passed as the closure to the create() method.

6.13.3.12 void Wombat::MamdaBasicSubscription::setTransport
(MamaTransport * transport)
Set the MAMA transport.
Do this before calling activate().

6.13.3.13 MamaTransport* Wombat::MamdaBasicSubscription::getTransport
() const
Get the MamaTransport for this subscription.

6.13.3.14 void Wombat::MamdaBasicSubscription::setQueue (MamaQueue * queue)
Set the MAMA queue.
Do this before calling activate().

6.13.3.15 MamaQueue* Wombat::MamdaBasicSubscription::getQueue () const
Get the MAMA queue for this subscription.
The documentation for this class was generated from the following file:

- MamdaBasicSubscription.h
6.14 Wombat::MamdaBookAtomicBookHandler

Class Reference

MamdaBookAtomicBookHandler is an interface for applications that need to know when a MamdaBookAtomicListener finishes processing a single book update.

#include <MamdaBookAtomicBookHandler.h>

Public Member Functions

- virtual void onBookAtomicBeginBook (MamdaSubscription *subscription, MamdaBookAtomicListener &listener, bool isRecap)=0
  
  *Method invoked before we start processing the first level in a message.*

- virtual void onBookAtomicEndBook (MamdaSubscription *subscription, MamdaBookAtomicListener &listener)=0
  
  *Method invoked when we stop processing the last level in a message.*

- virtual void onBookAtomicClear (MamdaSubscription *subscription, MamdaBookAtomicListener &listener, const MamaMsg &msg)=0
  
  *Method invoked when an order book is cleared.*

- virtual void onBookAtomicGap (MamdaSubscription *subscription, MamdaBookAtomicListener &listener, const MamaMsg &msg, const MamdaBookAtomicGap &event)=0
  
  *Method invoked when a gap in orderBook reports is discovered.*

- virtual ~MamdaBookAtomicBookHandler ()

6.14.1 Detailed Description

MamdaBookAtomicBookHandler is an interface for applications that need to know when a MamdaBookAtomicListener finishes processing a single book update.

This may be useful for applications that wish to destroy the subscription from a callback as the subscription can only be destroyed after the message processing is complete. Furthermore, it allows applications to determine when to clear the book when a recap arrives.
6.14 Wombat::MamdaBookAtomicBookHandler Class Reference

6.14.2 Constructor & Destructor Documentation

6.14.2.1 virtual Wombat::MamdaBookAtomicBookHandler::~MamdaBookAtomicBookHandler () [virtual]

100 {};

6.14.3 Member Function Documentation

6.14.3.1 virtual void Wombat::MamdaBookAtomicBookHandler::onBookAtomicBeginBook (MamdaSubscription * subscription, MamdaBookAtomicListener & listener, bool isRecap) [pure virtual]

Method invoked before we start processing the first level in a message.
The book should be cleared when isRecap == true.

Parameters:

subscription The MamdaSubscription handle.

listener The listener handling recaps/updates.

isRecap Whether the first update was a recap.

6.14.3.2 virtual void Wombat::MamdaBookAtomicBookHandler::onBookAtomicEndBook (MamdaSubscription * subscription, MamdaBookAtomicListener & listener) [pure virtual]

Method invoked when we stop processing the last level in a message.
We invoke this method after the last entry for the level gets processed. The subscription
may be destroyed from this callback.

Parameters:

subscription The MamdaSubscription handle.

listener The listener handling recaps/updates.

6.14.3.3 virtual void Wombat::MamdaBookAtomicBookHandler::onBookAtomicClear (MamdaSubscription * subscription, MamdaBookAtomicListener & listener, const MamaMsg & msg) [pure virtual]

Method invoked when an order book is cleared.
Parameters:

subscription The MamdaSubscription handle.
listener The listener handling recaps/updates.
msg The MamaMsg that triggered this invocation.

6.14.3.4 virtual void Wombat::MamdaBookAtomicBookHandler::onBookAtomicGap (MamdaSubscription * subscription,
MamdaBookAtomicListener & listener, const MamaMsg & msg, const
MamdaBookAtomicGap & event) [pure virtual]

Method invoked when a gap in orderBook reports is discovered.

Parameters:

subscription The MamdaSubscription handle.
listener The listener handling recaps/updates.
msg The MamaMsg that triggered this invocation.
event The gap value object.

The documentation for this class was generated from the following file:

- MamdaBookAtomicBookHandler.h
MamdaBookAtomicGap is an interface that provides access to order book atomic update gap related fields.

#include <MamdaBookAtomicGap.h>

Inheritance diagram for Wombat::MamdaBookAtomicGap:

```
Wombat::MamdaBasicEvent

Wombat::MamdaBookAtomicGap

Wombat::MamdaBookAtomicListener
```

Public Member Functions

- virtual mama_seqnum_t `getBeginGapSeqNum()` const =0
  
  The starting number of a detected sequence number gap in order book updates from the feedhandler.

- virtual mama_seqnum_t `getEndGapSeqNum()` const =0
  
  The ending number of a detected sequence number gap in order book updates from the feedhandler.

- virtual ~MamdaBookAtomicGap()

6.15.1 Detailed Description

MamdaBookAtomicGap is an interface that provides access to order book atomic update gap related fields.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 virtual Wombat::MamdaBookAtomicGap::~MamdaBookAtomicGap() [virtual]
6.15.3 Member Function Documentation

6.15.3.1 virtual mama_seqnum_t Wombat::MamdaBookAtomicGap::getBeginGapSeqNum () const [pure virtual]

The starting number of a detected sequence number gap in order book updates from the feedhandler.

Returns:

Starting sequence number of a sequence number gap.

Implemented in Wombat::MamdaBookAtomicListener.

6.15.3.2 virtual mama_seqnum_t Wombat::MamdaBookAtomicGap::getEndGapSeqNum () const [pure virtual]

The ending number of a detected sequence number gap in order book updates from the feedhandler.

Returns:

Ending sequence number of a sequence number gap.

Implemented in Wombat::MamdaBookAtomicListener.

The documentation for this class was generated from the following file:

- MamdaBookAtomicGap.h
MamdaBookAtomicLevel is an interface that provides access to trade related fields.

```cpp
#include <MamdaBookAtomicLevel.h>
```

Inheritance diagram for Wombat::MamdaBookAtomicLevel:

```
Wombat::MamdaBasicRecap

Wombat::MamdaBookAtomicLevel

Wombat::MamdaBookAtomicListener
```

**Public Member Functions**

- virtual mama_u32_t getPriceLevelNumLevels () const =0
  
  *Return the number of price levels in the order book update.*

- virtual mama_u32_t getPriceLevelNum () const =0
  
  *Return at which position this level is within an update containing a number of levels.*

- virtual double getPriceLevelPrice () const =0
  
  *Return the price for this price level.*

- virtual MamaPrice & getPriceLevelMamaPrice () const =0
  
  *Return the MamaPrice for this price level.*

- virtual mama_f64_t getPriceLevelSize () const =0
  
  *Return the number of order entries comprising this price level.*

- virtual mama_i64_t getPriceLevelSizeChange () const =0
  
  *Aggregate size at current price level.*

- virtual char getPriceLevelAction () const =0
  
  *The action to apply to the orderbook for this price level.*

- virtual char getPriceLevelSide () const =0
  
  *Side of book at current price level.*
• virtual const MamaDateTime & getPriceLevelTime () const =0
  
  Time of order book price level.

• virtual mama_f32_t getPriceLevelNumEntries () const =0
  
  Number of order book entries at current price level.

• virtual MamdaOrderBookTypes::OrderType getOrderType () const =0
  
  Get the Order Type.

• virtual void setOrderType (MamdaOrderBookTypes::OrderType orderType) const =0
  
  Set the Order Type.

• virtual ~MamdaBookAtomicLevel ()

6.16.1 Detailed Description

MamdaBookAtomicLevel is an interface that provides access to trade related fields.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 virtual Wombat::MamdaBookAtomicLevel::~MamdaBookAtomicLevel () [virtual]

148 {};

6.16.3 Member Function Documentation

6.16.3.1 virtual mama_u32_t Wombat::MamdaBookAtomicLevel::getPriceLevelNumLevels () const [pure virtual]

Return the number of price levels in the order book update.

Returns:

The number of price levels.

Implemented in Wombat::MamdaBookAtomicListener.
6.16.3.2 virtual mama_u32_t Wombat::MamdaBookAtomicLevel::getPriceLevelNum () const [pure virtual]

Return at which position this level is within an update containing a number of levels.
(i.e. level m of n levels in the update)

Returns:

The position of this level in the update received.

Implemented in Wombat::MamdaBookAtomicListener.

6.16.3.3 virtual double Wombat::MamdaBookAtomicLevel::getPriceLevelPrice () const [pure virtual]

Return the price for this price level.

Returns:

The price level price.

Implemented in Wombat::MamdaBookAtomicListener.

6.16.3.4 virtual MamaPrice& Wombat::MamdaBookAtomicLevel::getPriceLevelMamaPrice () const [pure virtual]

Return the MamaPrice for this price level.

Returns:

The price level price.

Implemented in Wombat::MamdaBookAtomicListener.

6.16.3.5 virtual mama_f64_t Wombat::MamdaBookAtomicLevel::getPriceLevelSize () const [pure virtual]

Return the number of order entries comprising this price level.

Warning:

Not supported for V5 entry book updates.
Returns:

The number of entries in this price level.

Implemented in `Wombat::MamdaBookAtomicListener`.

6.16.3.6 virtual mama_i64_t Wombat::MamdaBookAtomicLevel::getPriceLevelSizeChange () const [pure virtual]

Aggregate size at current price level.

Warning:

Not supported for V5 entry book updates.

Returns:

The aggregate size at the current price level.

Implemented in `Wombat::MamdaBookAtomicListener`.

6.16.3.7 virtual char Wombat::MamdaBookAtomicLevel::getPriceLevelAction () const [pure virtual]

The action to apply to the orderbook for this price level.
Can have a value of:

- A : Add a new price level
- U : Update an existing price level
- D : Delete an existing price level
- C : Closing information for price level (often treat the same as Update).

Warning:

Not supported for V5 entry book updates.

Returns:

The price level action.

Implemented in `Wombat::MamdaBookAtomicListener`. 
6.16.3.8 virtual char Wombat::MamdaBookAtomicLevel::getPriceLevelSide () const [pure virtual]

Side of book at current price level.

- B : Bid side. Same as 'buy' side.
- A : Ask side. Same as 'sell' side.

Returns:

The price level side.

Implemented in Wombat::MamdaBookAtomicListener.

6.16.3.9 virtual const MamaDateTime& Wombat::MamdaBookAtomicLevel::getPriceLevelTime () const [pure virtual]

Time of order book price level.

Returns:

The time of the orderbook price level.

Implemented in Wombat::MamdaBookAtomicListener.

6.16.3.10 virtual mama_f32_t Wombat::MamdaBookAtomicLevel::getPriceLevelNumEntries () const [pure virtual]

Number of order book entries at current price level.

Warning:

Not supported for V5 entry book updates.

Returns:

The number of entries at the current price level.

Implemented in Wombat::MamdaBookAtomicListener.
6.16.3.11 virtual MamdaOrderBookTypes::OrderType Wombat::MamdaBookAtomicLevel::getOrderType () const [pure virtual]

Get the Order Type.

**Returns:**

The order Type.

Implemented in Wombat::MamdaBookAtomicListener.

6.16.3.12 virtual void Wombat::MamdaBookAtomicLevel::setOrderType (MamdaOrderBookTypes::OrderType orderType) const [pure virtual]

Set the Order Type.

Implemented in Wombat::MamdaBookAtomicListener.

The documentation for this class was generated from the following file:

* MamdaBookAtomicLevel.h
MamdaBookAtomicLevel is an interface that provides access to Price Level and Price Level Entry fields.

#include <MamdaBookAtomicLevelEntry.h>

Inheritance diagram for Wombat::MamdaBookAtomicLevelEntry::

```
Wombat::MamdaBasicRecap
    ```
Wombat::MamdaBookAtomicLevelEntry
    ```
Wombat::MamdaBookAtomicListener
```

Public Member Functions

- virtual double getPriceLevelPrice() const =0
- virtual MamaPrice & getPriceLevelMamaPrice() const =0
- virtual mama_f64_t getPriceLevelSize() const =0
- virtual char getPriceLevelAction() const =0
- virtual char getPriceLevelSide() const =0
- virtual const MamaDateTime & getPriceLevelTime() const =0
- virtual mama_f32_t getPriceLevelNumEntries() const =0
- virtual mama_u32_t getPriceLevelActNumEntries() const =0
- virtual char getPriceLevelEntryAction() const =0
- Order book entry action to apply to the full order book.
- virtual char getPriceLevelEntryReason() const =0
  Order book entry reason.
- virtual const char * getPriceLevelEntryId() const =0
  Order book entry Id.
- virtual mama_u64_t getPriceLevelEntrySize() const =0
  Return the order book entry size.
- virtual const MamaDateTime & getPriceLevelEntryTime() const =0
  Return the time of order book entry update.
6.17.1 Detailed Description

*MamdaBookAtomicLevel* is an interface that provides access to Price Level and Price Level Entry fields.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 virtual Wombat::MamdaBookAtomicLevelEntry::~MamdaBookAtomicLevelEntry () [virtual]

119 {};

6.17.3 Member Function Documentation

6.17.3.1 virtual double Wombat::MamdaBookAtomicLevelEntry::getPriceLevelPrice () const [pure virtual]

See also:

*MamdaBookAtomicLevel::getPriceLevelPrice()*

Implemented in *Wombat::MamdaBookAtomicListener*.

6.17.3.2 virtual MamaPrice& Wombat::MamdaBookAtomicLevelEntry::getPriceLevelMamaPrice () const [pure virtual]

See also:

*MamdaBookAtomicLevel::getPriceLevelMamaPrice()*

Implemented in *Wombat::MamdaBookAtomicListener*.

6.17.3.3 virtual mama_f64_t Wombat::MamdaBookAtomicLevelEntry::getPriceLevelSize () const [pure virtual]

See also:

*MamdaBookAtomicLevel::getPriceLevelSize()*
6.17 Wombat::MamdaBookAtomicLevelEntry Class Reference

Implemented in Wombat::MamdaBookAtomicListener.

### 6.17.3.4 virtual char Wombat::MamdaBookAtomicLevelEntry::getPriceLevelAction () const [pure virtual]

See also:
- MamdaBookAtomicLevel::getPriceLevelAction()

Implemented in Wombat::MamdaBookAtomicListener.

### 6.17.3.5 virtual char Wombat::MamdaBookAtomicLevelEntry::getPriceLevelSide () const [pure virtual]

See also:
- MamdaBookAtomicLevel::getPriceLevelSide()

Implemented in Wombat::MamdaBookAtomicListener.

### 6.17.3.6 virtual const MamaDateTime& Wombat::MamdaBookAtomicLevelEntry::getPriceLevelTime () const [pure virtual]

See also:
- MamdaBookAtomicLevel::getPriceLevelTime()

Implemented in Wombat::MamdaBookAtomicListener.

### 6.17.3.7 virtual mama_f32_t Wombat::MamdaBookAtomicLevelEntry::getPriceLevelNumEntries () const [pure virtual]

See also:
- MamdaBookAtomicLevel::getPriceLevelNumEntries()

Implemented in Wombat::MamdaBookAtomicListener.

### 6.17.3.8 virtual mama_u32_t Wombat::MamdaBookAtomicLevelEntry::getPriceLevelActNumEntries () const [pure virtual]

Implemented in Wombat::MamdaBookAtomicListener.
6.17.3.9 virtual char Wombat::MamdaBookAtomicLevelEntry::getPriceLevelEntryAction () const  [pure virtual]

Order book entry action to apply to the full order book.

- A: Add entry to the price level.
- U: Update existing entry in the price level.
- D: Delete existing entry from the price level.

Returns:

The orderbook entry action.

Implemented in Wombat::MamdaBookAtomicListener.

6.17.3.10 virtual char Wombat::MamdaBookAtomicLevelEntry::getPriceLevelEntryReason () const  [pure virtual]

Order book entry reason.

Returns:

The orderbook entry reason.

Implemented in Wombat::MamdaBookAtomicListener.

6.17.3.11 virtual const char* Wombat::MamdaBookAtomicLevelEntry::getPriceLevelEntryId () const  [pure virtual]

Order book entry Id.
(order ID, participant ID, etc.)

Returns:

The entry id

Implemented in Wombat::MamdaBookAtomicListener.
virtual mama_u64_t Wombat::MamdaBookAtomicLevelEntry::getPriceLevelEntrySize () const [pure virtual]

Return the order book entry size.

Returns:
Order book entry size

Implemented in Wombat::MamdaBookAtomicListener.

virtual const MamaDateTime& Wombat::MamdaBookAtomicLevelEntry::getPriceLevelEntryTime () const [pure virtual]

Return the time of order book entry update.

Returns:
Time of order book entry update.

Implemented in Wombat::MamdaBookAtomicListener.

The documentation for this class was generated from the following file:

- MamdaBookAtomicLevelEntry.h
6.18 Wombat::MamdaBookAtomicLevelEntryHandler Class Reference

MamdaBookAtomicLevelEntryHandler is an interface for applications that want to have an easy way to handle order book Price Level & Entry updates.

#include <MamdaBookAtomicLevelEntryHandler.h>

Public Member Functions

• virtual void onBookAtomicLevelEntryRecap (MamdaSubscription *subscription, MamdaBookAtomicListener &listener, const MamaMsg &msg, const MamdaBookAtomicLevelEntry &levelEntry)=0

  Method invoked when a full refresh of the order book for the security is available.

• virtual void onBookAtomicLevelEntryDelta (MamdaSubscription *subscription, MamdaBookAtomicListener &listener, const MamaMsg &msg, const MamdaBookAtomicLevelEntry &levelEntry)=0

  Method invoked when an order book delta is reported.

• virtual ~MamdaBookAtomicLevelEntryHandler ()

6.18.1 Detailed Description

MamdaBookAtomicLevelEntryHandler is an interface for applications that want to have an easy way to handle order book Price Level & Entry updates.

The interface defines callback methods for different types of orderBook-related events: order book recaps and updates.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 virtual Wombat::MamdaBookAtomicLevelEntryHandler::~MamdaBookAtomicLevelEntryHandler ()

[virtual]
6.18 Wombat::MamdaBookAtomicLevelEntryHandler Class Reference

6.18.3 Member Function Documentation

6.18.3.1 virtual void Wombat::MamdaBookAtomicLevelEntryHandler::onBookAtomicLevelEntryRecap (MamdaSubscription * subscription, MamdaBookAtomicListener & listener, const MamaMsg & msg, const MamdaBookAtomicLevelEntry & levelEntry) [pure virtual]

Method invoked when a full refresh of the order book for the security is available.
The reason for the invocation may be any of the following:

- Initial image.
- Recap update (e.g., after server fault tolerant event or data quality event.)
- After stale status removed.

Parameters:

subscription The MamdaSubscription handle.
listener The listener handling the recap.
msg The MamaMsg that triggered this invocation.
levelEntry The Price Level Entry recap.

6.18.3.2 virtual void Wombat::MamdaBookAtomicLevelEntryHandler::onBookAtomicLevelEntryDelta (MamdaSubscription * subscription, MamdaBookAtomicListener & listener, const MamaMsg & msg, const MamdaBookAtomicLevelEntry & levelEntry) [pure virtual]

Method invoked when an order book delta is reported.

Parameters:

subscription The MamdaSubscription handle.
listener The listener handling the update.
msg The MamaMsg that triggered this invocation.
levelEntry The Price Level Entry update.

The documentation for this class was generated from the following file:

- MamdaBookAtomicLevelEntryHandler.h
6.19 Wombat::MamdaBookAtomicLevelHandler
Class Reference

MamdaBookAtomicLevelHandler is an interface for applications that want to have an easy way to handle order book Price Level updates.

#include <MamdaBookAtomicLevelHandler.h>

Public Member Functions

- virtual void onBookAtomicLevelRecap (MamdaSubscription *subscription, MamdaBookAtomicListener &listener, const MamaMsg &msg, const MamdaBookAtomicLevel &level)=0

  Method invoked when a full refresh of the order book for the security is available.

- virtual void onBookAtomicLevelDelta (MamdaSubscription *subscription, MamdaBookAtomicListener &listener, const MamaMsg &msg, const MamdaBookAtomicLevel &level)=0

  Method invoked when an order book delta is reported.

- virtual ~MamdaBookAtomicLevelHandler ()

6.19.1 Detailed Description

MamdaBookAtomicLevelHandler is an interface for applications that want to have an easy way to handle order book Price Level updates.

The interface defines callback methods for different types of orderBook-related events: order book recaps and updates.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 virtual Wombat::MamdaBookAtomicLevelHandler::~MamdaBookAtomicLevelHandler () [virtual]

80 {};

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.19.3 Member Function Documentation

6.19.3.1 virtual void Wombat::MamdaBookAtomicLevelHandler::onBookAtomicLevelRecap (MamdaSubscription ∗ subscription, MamdaBookAtomicListener & listener, const MamaMsg & msg, const MamdaBookAtomicLevel & level) [pure virtual]

Method invoked when a full refresh of the order book for the security is available.

The reason for the invocation may be any of the following:

- Initial image.
- Recap update (e.g., after server fault tolerant event or data quality event.)
- After stale status removed.

Parameters:

subscription  The MamdaSubscription handle.

listener  The listener handling the recap.

msg  The MamaMsg that triggered this invocation.

level  The Price Level recap.

6.19.3.2 virtual void Wombat::MamdaBookAtomicLevelHandler::onBookAtomicLevelDelta (MamdaSubscription ∗ subscription, MamdaBookAtomicListener & listener, const MamaMsg & msg, const MamdaBookAtomicLevel & level) [pure virtual]

Method invoked when an order book delta is reported.

Parameters:

subscription  The MamdaSubscription handle.

listener  The listener handling the update.

msg  The MamaMsg that triggered this invocation.

level  The Price Level update.

The documentation for this class was generated from the following file:

- MamdaBookAtomicLevelHandler.h
MamdaBookAtomicListener is a class that specializes in handling order book updates.

```cpp
#include <MamdaBookAtomicListener.h>
```

Inheritance diagram for Wombat::MamdaBookAtomicListener:

```
Wombat::MamdaBasicRecap Wombat::MamdaBasicRecap
Wombat::MamdaBasicRecap Wombat::MamdaBasicEvent
Wombat::MamdaMsgListener Wombat::MamdaBookAtomicLevel Wombat::MamdaBookAtomicLevelEntry Wombat::MamdaBookAtomicGap
Wombat::MamdaBasicRecap Wombat::MamdaBasicRecap Wombat::MamdaBasicEvent
Wombat::MamdaBasicRecap Wombat::MamdaBasicRecap Wombat::MamdaBasicEvent
```

### Public Member Functions

- `MamdaBookAtomicListener ()
- virtual ~MamdaBookAtomicListener ()
- void addBookHandler (MamdaBookAtomicBookHandler *handler)
- void addLevelHandler (MamdaBookAtomicLevelHandler *handler)
- void addLevelEntryHandler (MamdaBookAtomicLevelEntryHandler *handler)
- `const char * getSymbol () const
  Get the instruments string symbol.
- `const char * getPartId () const
  Get the participant identifier.
- `const MamaDateTime & getSrcTime () const
  Get the source time.
- `const MamaDateTime & getActivityTime () const
  Get the activity time.
- `const MamaDateTime & getLineTime () const
  Get the line time.
- `const MamaDateTime & getSendTime () const
  Get the send time.
- `const MamaMsgQual & getMsgQual () const
  Get the message qualifier.
• const MamaDateTime & getEventTime () const
  Get the event time.

• mama_seqnum_t getEventSeqNum () const
  Get the event sequence number.

• MamdaFieldState getSymbolFieldState () const
  Get the string symbol field state.

• MamdaFieldState getPartIdFieldState () const
  Get the participant identifier field state.

• MamdaFieldState getEventSeqNumFieldState () const
  Get the event sequence number field state.

• MamdaFieldState getEventTimeFieldState () const
  Get the event time field state.

• MamdaFieldState getSrCtimeFieldState () const
  Get the source time field state.

• MamdaFieldState getActivityTimeFieldState () const
  Get the activity time field state.

• MamdaFieldState getLineTimeFieldState () const
  Get the line time of the update.

• MamdaFieldState getSendTimeFieldState () const
  Get the send time field state.

• MamdaFieldState getMsgQualFieldState () const
  Get the message qualifier field state.

• mama_u32_t getPriceLevelNumLevels () const
  Return the number of price levels in the order book update.

• mama_u32_t getPriceLevelNum () const
  Return at which position this level is within an update containing a number of levels.

• double getPriceLevelPrice () const
  Return the price for this price level.
• MamaPrice & getPriceLevelMamaPrice() const
  Return the MamaPrice for this price level.

• mama_f64_t getPriceLevelSize() const
  Return the number of order entries comprising this price level.

• mama_i64_t getPriceLevelSizeChange() const
  Aggregate size at current price level.

• char getPriceLevelAction() const
  The action to apply to the orderbook for this price level.

• char getPriceLevelSide() const
  Side of book at current price level.

• const MamaDateTime & getPriceLevelTime() const
  Time of order book price level.

• mama_f32_t getPriceLevelNumEntries() const
  Number of order book entries at current price level.

• mama_u32_t getPriceLevelActNumEntries() const
  Order book entry action to apply to the full order book.

• char getPriceLevelEntryReason() const
  Order book entry reason.

• const char * getPriceLevelEntryId() const
  Order book entry Id.

• mama_u64_t getPriceLevelEntrySize() const
  Return the order book entry size.

• const MamaDateTime & getPriceLevelEntryTime() const
  Return the time of order book entry update.

• MamdaOrderBookTypes::OrderType getOrderType() const
  Get the Order Type.

• void setOrderType(MamdaOrderBookTypes::OrderType orderType) const
  Set the Order Type.
• `bool getHasMarketOrders () const`

• `void setProcessMarketOrders (bool process) const`

• `mama_seqnum_t getBeginGapSeqNum () const`

   The starting number of a detected sequence number gap in order book updates from
   the feedhandler.

• `mama_seqnum_t getEndGapSeqNum () const`

   The ending number of a detected sequence number gap in order book updates from
   the feedhandler.

• `virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)`

### 6.20.1 Detailed Description

*MamdaBookAtomicListener* is a class that specializes in handling order book updates. Unlike the *MamdaOrderBookListener* no actual order book is built or maintained. The sole purpose of this is to provide clients direct access to the orderbook updates without the overhead of maintaining a book. Developers provide their own implementation of either or both the *MamdaBookAtomicLevelHandler* and the *MamdaBookAtomicLevelEntryHandler* interfaces and will be delivered notifications for order book recaps and deltas. While the *MamdaBookAtomicLevelHandler* handles recaps and deltas at a Price Level granularity the *MamdaBookAtomicLevelEntryHandler* handles recaps and deltas at a Price Level Entry level (both level & entry data). Notifications for order book deltas include only the delta. An obvious application for this MAMDA class is any kind of program trading application that needs to build its own order book or an application that needs to archive order book data.

Note: The *MamdaBookAtomicListener* class caches only order book deltas. If only an *MamdaBookAtomicLevelHandler* is add to this listener updates and deltas are only processed to Price Level granularity and Entry Level data ignored saving on processing time. Among other reasons, caching of these fields makes it possible to provide complete trade-related callbacks, even when the publisher (e.g., feed handler) is only publishing deltas containing modified fields.
6.20.2 Constructor & Destructor Documentation

6.20.2.1 Wombat::MamdaBookAtomicListener::MamdaBookAtomicListener()

6.20.2.2 virtual Wombat::MamdaBookAtomicListener::~MamdaBookAtomicListener() [virtual]

6.20.3 Member Function Documentation

6.20.3.1 void Wombat::MamdaBookAtomicListener::addBookHandler(MamdaBookAtomicBookHandler * handler)

6.20.3.2 void Wombat::MamdaBookAtomicListener::addLevelHandler(MamdaBookAtomicLevelHandler * handler)

6.20.3.3 void Wombat::MamdaBookAtomicListener::addLevelEntryHandler(MamdaBookAtomicLevelEntryHandler * handler)

6.20.3.4 const char* Wombat::MamdaBookAtomicListener::getSymbol() const [virtual]

Get the instruments string symbol.

Returns:

Symbol. This is the "well-known" symbol for the security, including any symbology mapping performed by the publisher.

Implements Wombat::MamdaBasicEvent.

6.20.3.5 const char* Wombat::MamdaBookAtomicListener::getPartId() const [virtual]

Get the participant identifier.

Returns:

Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).

Implements Wombat::MamdaBasicEvent.
6.20 Wombat::MamdaBookAtomicListener Class Reference

6.20.3.6 const MamaDateTime& Wombat::MamdaBookAtomicListener::getSrcTime () const [virtual]

Get the source time.

**Returns:**

Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.20.3.7 const MamaDateTime& Wombat::MamdaBookAtomicListener::getActivityTime () const [virtual]

Get the activity time.

**Returns:**

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements Wombat::MamdaBasicEvent.

6.20.3.8 const MamaDateTime& Wombat::MamdaBookAtomicListener::getLineTime () const [virtual]

Get the line time.

**Returns:**

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

6.20.3.9 const MamaDateTime& Wombat::MamdaBookAtomicListener::getSendTime () const [virtual]

Get the send time.

---

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
Returns:
Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime()).

Implements Wombat::MamdaBasicEvent.

6.20.3.10 const MamaMsgQual& Wombat::MamdaBookAtomicListener::getMsgQual () const [virtual]

Get the message qualifier.

Returns:
Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.20.3.11 const MamaDateTime& Wombat::MamdaBookAtomicListener::getEventTime () const [virtual]

Get the event time.

Returns:
Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.20.3.12 mama_seqnum_t Wombat::MamdaBookAtomicListener::getEventSeqNum () const [virtual]

Get the event sequence number.

Returns:
Source sequence number. The exchange generated sequence number.

Implements Wombat::MamdaBasicEvent.
6.20 Wombat::MamdaBookAtomicListener Class Reference

6.20.3.13 MamdaFieldState Wombat::MamdaBookAtomicListener::getSymbolFieldState () const [virtual]

Get the string symbol field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.20.3.14 MamdaFieldState Wombat::MamdaBookAtomicListener::getPartIdFieldState () const [virtual]

Get the participant identifier field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.20.3.15 MamdaFieldState Wombat::MamdaBookAtomicListener::getEventSeqNumFieldState () const [virtual]

Get the event sequence number field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.20.3.16 MamdaFieldState Wombat::MamdaBookAtomicListener::getEventTimeFieldState () const [virtual]

Get the event time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.20.3.17 MamdaFieldState Wombat::MamdaBookAtomicListener::getSourceTimeFieldState () const [virtual]

Get the source time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.20.3.18 MamdaFieldState Wombat::MamdaBookAtomicListener::getActivityTimeFieldState () const [virtual]

Get the activity time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.20.3.19 MamdaFieldState Wombat::MamdaBookAtomicListener::getLineTimeFieldState () const [virtual]

Get the line time of the update.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.20.3.20 MamdaFieldState Wombat::MamdaBookAtomicListener::getSendTimeFieldState () const [virtual]

Get the send time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.20 Wombat::MamdaBookAtomicListener Class Reference

6.20.3.21 MamdaFieldState Wombat::MamdaBookAtomicListener::getMsgQualFieldState () const [virtual]

Get the message qualifier field state.

**Returns:**

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.20.3.22 mama_u32_t Wombat::MamdaBookAtomicListener::getPriceLevelNumLevels () const [virtual]

Return the number of price levels in the order book update.

**Returns:**

The number of price levels.

Implements Wombat::MamdaBookAtomicLevel.

6.20.3.23 mama_u32_t Wombat::MamdaBookAtomicListener::getPriceLevelNum () const [virtual]

Return at which position this level is within an update containing a number of levels. (i.e. level m of n levels in the update)

**Returns:**

The position of this level in the update received.

Implements Wombat::MamdaBookAtomicLevel.

6.20.3.24 double Wombat::MamdaBookAtomicListener::getPriceLevelPrice () const [virtual]

Return the price for this price level.

**Returns:**

The price level price.

Implements Wombat::MamdaBookAtomicLevel.
6.20.3.25 MamaPrice& Wombat::MamdaBookAtomicListener::getPriceLevel-
MamaPrice () const [virtual]

Return the MamaPrice for this price level.

Returns:

The price level price.

Implements Wombat::MamdaBookAtomicLevel.

6.20.3.26 mama_f64_t Wombat::MamdaBookAtomicListener::getPriceLevel-
Size () const [virtual]

Return the number of order entries comprising this price level.

Warning:

Not supported for V5 entry book updates.

Returns:

The number of entries in this price level.

Implements Wombat::MamdaBookAtomicLevel.

6.20.3.27 mama_i64_t Wombat::MamdaBookAtomicListener::getPriceLevel-
SizeChange () const [virtual]

Aggregate size at current price level.

Warning:

Not supported for V5 entry book updates.

Returns:

The aggregate size at the current price level.

Implements Wombat::MamdaBookAtomicLevel.

6.20.3.28 char Wombat::MamdaBookAtomicListener::getPriceLevelAction ()
const [virtual]

The action to apply to the orderbook for this price level.

Can have a value of:
6.20 Wombat::MamdaBookAtomicListener Class Reference

- A : Add a new price level
- U : Update an existing price level
- D : Delete an existing price level
- C : Closing information for price level (often treat the same as Update).

**Warning:**

Not supported for V5 entry book updates.

**Returns:**

The price level action.

Implements Wombat::MamdaBookAtomicLevel.

6.20.3.29 char Wombat::MamdaBookAtomicListener::getPriceLevelSide () const [virtual]

Side of book at current price level.

- B : Bid side. Same as 'buy' side.
- A : Ask side. Same as 'sell' side.

**Returns:**

The price level side.

Implements Wombat::MamdaBookAtomicLevel.

6.20.3.30 const MamaDateTime& Wombat::MamdaBookAtomicListener::getPriceLevelTime () const [virtual]

Time of order book price level.

**Returns:**

The time of the orderbook price level.

Implements Wombat::MamdaBookAtomicLevel.
6.20.3.31 mama_f32_t Wombat::MamdaBookAtomicListener::getPriceLevelNumEntries () const [virtual]

Number of order book entries at current price level.

Warning:

Not supported for V5 entry book updates.

Returns:

The number of entries at the current price level.

Implements Wombat::MamdaBookAtomicLevel.

6.20.3.32 mama_u32_t Wombat::MamdaBookAtomicListener::getPriceLevelActNumEntries () const [virtual]

Implements Wombat::MamdaBookAtomicLevelEntry.

6.20.3.33 char Wombat::MamdaBookAtomicListener::getPriceLevelEntryAction () const [virtual]

Order book entry action to apply to the full order book.

- A: Add entry to the price level.
- U: Update existing entry in the price level.
- D: Delete existing entry from the price level.

Returns:

The orderbook entry action.

Implements Wombat::MamdaBookAtomicLevelEntry.

6.20.3.34 char Wombat::MamdaBookAtomicListener::getPriceLevelEntryReason () const [virtual]

Order book entry reason.

Returns:

The orderbook entry reason.

Implements Wombat::MamdaBookAtomicLevelEntry.
6.20.3.35  const char* Wombat::MamdaBookAtomicListener::getPriceLevelEntryId() const  [virtual]

Order book entry Id.
(order ID, participant ID, etc.)

Returns:

The entry id

Implements Wombat::MamdaBookAtomicLevelEntry.

6.20.3.36  mama_u64_t Wombat::MamdaBookAtomicListener::getPriceLevelEntrySize() const  [virtual]

Return the order book entry size.

Returns:

Order book entry size

Implements Wombat::MamdaBookAtomicLevelEntry.

6.20.3.37  const MamaDateTime& Wombat::MamdaBookAtomicListener::getPriceLevelEntryTime() const  [virtual]

Return the time of order book entry update.

Returns:

Time of order book entry update.

Implements Wombat::MamdaBookAtomicLevelEntry.

6.20.3.38  MamdaOrderBookTypes::OrderType Wombat::MamdaBookAtomicListener::getOrderType() const  [virtual]

Get the Order Type.

Returns:

The order Type.

Implements Wombat::MamdaBookAtomicLevel.
6.20.3.39  void Wombat::MamdaBookAtomicListener::setOrderType (
MamdaOrderBookTypes::OrderType orderType) const
  [virtual]

Set the Order Type.
Implements Wombat::MamdaBookAtomicLevel.

6.20.3.40  bool Wombat::MamdaBookAtomicListener::getHasMarketOrders () const

6.20.3.41  void Wombat::MamdaBookAtomicListener::setProcessMarket-
  Orders (bool process) const

6.20.3.42  mama_seqnum_t Wombat::MamdaBookAtomicListener::getBegin-
  GapSeqNum () const  [virtual]

The starting number of a detected sequence number gap in order book updates from
the feedhandler.

  Returns:

  Starting sequence number of a sequence number gap.

Implements Wombat::MamdaBookAtomicGap.

6.20.3.43  mama_seqnum_t Wombat::MamdaBookAtomicListener::getEnd-
  GapSeqNum () const  [virtual]

The ending number of a detected sequence number gap in order book updates from the
feedhandler.

  Returns:

  Ending sequence number of a sequence number gap.

Implements Wombat::MamdaBookAtomicGap.

6.20.3.44  virtual void Wombat::MamdaBookAtomicListener::onMsg
  (MamdaSubscription * subscription, const MamaMsg & msg, short
  msgType)  [virtual]

Implements Wombat::MamdaMsgListener.

The documentation for this class was generated from the following file:
6.20 Wombat::MamdaBookAtomicListener Class Reference

- MamdaBookAtomicListener.h
6.21 Wombat::MamdaCheckerHandler Class Reference

MamdaCheckerHandler is an interface for applications that want to handle the results of the MamdaQuoteChecker and MamdaTradeChecker.

#include <MamdaCheckerHandler.h>

Public Member Functions

- virtual void onSuccess (MamdaCheckerType checkType)=0
  Method invoked when a successful check is completed.

- virtual void onInconclusive (MamdaCheckerType checkType, const char *reason)=0
  Method invoked when check is completed inconclusively.

- virtual void onFailure (MamdaCheckerType checkType, const char *reason, const MamaMsg &msg)=0
  Method invoked when a failed check is completed.

6.21.1 Detailed Description

MamdaCheckerHandler is an interface for applications that want to handle the results of the MamdaQuoteChecker and MamdaTradeChecker. Callback interfaces are provided for correct and erroneous checks.

6.21.2 Member Function Documentation

6.21.2.1 virtual void Wombat::MamdaCheckerHandler::onSuccess (MamdaCheckerType checkType) [pure virtual]

Method invoked when a successful check is completed.

6.21.2.2 virtual void Wombat::MamdaCheckerHandler::onInconclusive (MamdaCheckerType checkType, const char *reason) [pure virtual]

Method invoked when check is completed inconclusively.
An attempt to check the order book may be inconclusive if the order book sequence numbers do not match up.

### 6.21.2.3 virtual void Wombat::MamdaCheckerHandler::onFailure

```cpp
(MamdaCheckerType checkType, const char * reason, const MamaMsg & msg) [pure virtual]
```

Method invoked when a failed check is completed.

The message provided, if non-NULL, is the one received for the snapshot or delta, depending upon the value of checkType.

The documentation for this class was generated from the following file:

- MamdaCheckerHandler.h
6.22 Wombat::MamdaCommonFields Class Reference

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing common fields from update messages.

```cpp
#include <MamdaCommonFields.h>
```

### Static Public Member Functions

- static void `setDictionary` (const MamaDictionary &dictionary)
- static void `reset` ()

  Reset the dictionary for common update fields.

- static bool `isSet` ()

### Static Public Attributes

- static const MamaFieldDescriptor * `SYMBOL`
- static const MamaFieldDescriptor * `ISSUE_SYMBOL`
- static const MamaFieldDescriptor * `INDEX_SYMBOL`
- static const MamaFieldDescriptor * `PART_ID`
- static const MamaFieldDescriptor * `SEQ_NUM`
- static const MamaFieldDescriptor * `SRC_TIME`
- static const MamaFieldDescriptor * `LINE_TIME`
- static const MamaFieldDescriptor * `ACTIVITY_TIME`
- static const MamaFieldDescriptor * `SEND_TIME`
- static const MamaFieldDescriptor * `PUB_ID`
- static const MamaFieldDescriptor * `MSG_QUAL`
- static const MamaFieldDescriptor * `MSG_SEQ_NUM`
- static const MamaFieldDescriptor * `MSG_NUM`
- static const MamaFieldDescriptor * `MSG_TOTAL`
- static const MamaFieldDescriptor * `INITIAL_TOTAL`
- static const MamaFieldDescriptor * `SENDER_ID`

### 6.22.1 Detailed Description

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing common fields from update messages.

This class should be initialized prior to using the listener objects by calling `MamdaCommonFields.setDictionary()` with a valid dictionary object which contains these common fields.
6.22.2 Member Function Documentation

6.22.2.1 static void Wombat::MamdaCommonFields::setDictionary (const MamaDictionary & dictionary) [static]

6.22.2.2 static void Wombat::MamdaCommonFields::reset () [static]

Reset the dictionary for common update fields.
6.22 Wombat::MamdaCommonFields Class Reference

6.22.3 static bool Wombat::MamdaCommonFields::isSet () [static]

6.22.3 Member Data Documentation

6.22.3.1 const MamaFieldDescriptor* Wombat::MamdaCommonFields::SYMBOL [static]

6.22.3.2 const MamaFieldDescriptor* Wombat::MamdaCommonFields::ISSUE_SYMBOL [static]

6.22.3.3 const MamaFieldDescriptor* Wombat::MamdaCommonFields::INDEX_SYMBOL [static]

6.22.3.4 const MamaFieldDescriptor* Wombat::MamdaCommonFields::PART_ID [static]

6.22.3.5 const MamaFieldDescriptor* Wombat::MamdaCommonFields::SEQ_NUM [static]

6.22.3.6 const MamaFieldDescriptor* Wombat::MamdaCommonFields::SRC_TIME [static]

6.22.3.7 const MamaFieldDescriptor* Wombat::MamdaCommonFields::LINE_TIME [static]

6.22.3.8 const MamaFieldDescriptor* Wombat::MamdaCommonFields::ACTIVITY_TIME [static]

6.22.3.9 const MamaFieldDescriptor* Wombat::MamdaCommonFields::SEND_TIME [static]

6.22.3.10 const MamaFieldDescriptor* Wombat::MamdaCommonFields::PUB_ID [static]

6.22.3.11 const MamaFieldDescriptor* Wombat::MamdaCommonFields::MSG_QUAL [static]

6.22.3.12 const MamaFieldDescriptor* Wombat::MamdaCommonFields::MSG_SEQ_NUM [static]

6.22.3.13 const MamaFieldDescriptor* Wombat::MamdaCommonFields::MSG_NUM [static]

6.22.3.14 const MamaFieldDescriptor* Wombat::MamdaCommonFields::MSG_TOTAL [static]

6.22.3.15 const MamaFieldDescriptor* Wombat::MamdaCommonFields::INITIAL_TOTAL [static]

6.22.3.16 const MamaFieldDescriptor* Wombat::MamdaCommonFields::SENDER_ID [static]

The documentation for this class was generated from the following file:
• MamdaCommonFields.h
MamdaConcreteBasicEvent is intended to be used to help implement concrete versions of various classes derived from MamdaBasicEvent.

#include <MamdaConcreteBasicEvent.h>

Inheritance diagram for Wombat::MamdaConcreteBasicEvent::

```
Wombat::MamdaBasicEvent
    ▼
Wombat::MamdaConcreteBasicEvent
```

**Public Member Functions**

- `MamdaConcreteBasicEvent ()`
  
  *Constructor.*

- `virtual ~MamdaConcreteBasicEvent ()`
  
  *Destructor.*

- `virtual const char * getSymbol () const`
  
  *Get the instruments string symbol.*

- `virtual const char * getPartId () const`
  
  *Get the participant identifier.*

- `virtual mama_seqnum_t getEventSeqNum () const`
  
  *Get the event sequence number.*

- `virtual const MamaDateTime & getEventTime () const`
  
  *Get the event time.*

- `virtual const MamaDateTime & getSrcTime () const`
  
  *Get the source time.*

- `virtual const MamaDateTime & getActivityTime () const`
  
  *Get the activity time.*

- `virtual const MamaDateTime & getLineTime () const`
Get the line time.

- virtual const MamaDateTime & getSendTime () const
  
  Get the send time.

- virtual const MamaMsgQual & getMsgQual () const
  
  Get the message qualifier.

- virtual MamdaFieldState getSymbolFieldState () const
  
  Get the string symbol field state.

- virtual MamdaFieldState getPartIdFieldState () const
  
  Get the participant identifier field state.

- virtual MamdaFieldState getEventSeqNumFieldState () const
  
  Get the event sequence number field state.

- virtual MamdaFieldState getEventTimeFieldState () const
  
  Get the event time field state.

- virtual MamdaFieldState getSrcTimeFieldState () const
  
  Get the source time field state.

- virtual MamdaFieldState getActivityTimeFieldState () const
  
  Get the activity time field state.

- virtual MamdaFieldState getLineTimeFieldState () const
  
  Get the line time of the update.

- virtual MamdaFieldState getSendTimeFieldState () const
  
  Get the send time field state.

- virtual MamdaFieldState getMsgQualFieldState () const
  
  Get the message qualifier field state.

- virtual void setSymbol (const char ∗value)
- virtual void setPartId (const char ∗value)
- virtual void setEventSeqNum (mama_seqnum_t value)
- virtual void setEventTime (const MamaDateTime &value)
- virtual void setSrcTime (const MamaDateTime &value)
- virtual void setActivityTime (const MamaDateTime &value)
- virtual void setLineTime (const MamaDateTime &value)
- virtual void setSendTime (const MamaDateTime &value)
- virtual void setMsgQual (const MamaMsgQual &value)
6.23 Wombat::MamdaConcreteBasicEvent Class Reference

6.23.1 Detailed Description

MamdaConcreteBasicEvent is intended to be used to help implement concrete versions of various classes derived from MamdaBasicEvent.

6.23.2 Constructor & Destructor Documentation

6.23.2.1 Wombat::MamdaConcreteBasicEvent::MamdaConcreteBasicEvent ()
Constructor.

6.23.2.2 virtual Wombat::MamdaConcreteBasicEvent::~MamdaConcreteBasicEvent () [virtual]
Destructor.

6.23.3 Member Function Documentation

6.23.3.1 virtual const char* Wombat::MamdaConcreteBasicEvent::getSymbol () const [virtual]
Get the instruments string symbol.

Returns:
Symbol. This is the "well-known" symbol for the security, including any symbology mapping performed by the publisher.

Implements Wombat::MamdaBasicEvent.

6.23.3.2 virtual const char* Wombat::MamdaConcreteBasicEvent::getPartId () const [virtual]
Get the participant identifier.

Returns:
Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).

Implements Wombat::MamdaBasicEvent.

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.23.3.3  virtual mama_seqnum_t Wombat::MamdaConcreteBasicEvent::getEventSeqNum () const [virtual]

Get the event sequence number.

**Returns:**

Source sequence number. The exchange generated sequence number.

Implements **Wombat::MamdaBasicEvent**.

6.23.3.4  virtual const MamaDateTime& Wombat::MamdaConcreteBasicEvent::getEventTime () const [virtual]

Get the event time.

**Returns:**

Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements **Wombat::MamdaBasicEvent**.

6.23.3.5  virtual const MamaDateTime& Wombat::MamdaConcreteBasicEvent::getSrcTime () const [virtual]

Get the source time.

**Returns:**

Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements **Wombat::MamdaBasicEvent**.

6.23.3.6  virtual const MamaDateTime& Wombat::MamdaConcreteBasicEvent::getActivityTime () const [virtual]

Get the activity time.
Returns:

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements `Wombat::MamdaBasicEvent`.

### 6.23.3.7 virtual const MamaDateTime& `Wombat::MamdaConcreteBasicEvent::getLineTime () const [virtual]`

Get the line time.

Returns:

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements `Wombat::MamdaBasicEvent`.

### 6.23.3.8 virtual const MamaDateTime& `Wombat::MamdaConcreteBasicEvent::getSendTime () const [virtual]`

Get the send time.

Returns:

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime()).

Implements `Wombat::MamdaBasicEvent`.

### 6.23.3.9 virtual const MamaMsgQual& `Wombat::MamdaConcreteBasicEvent::getMsgQual () const [virtual]`

Get the message qualifier.

Returns:

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.
Implements Wombat::MamdaBasicEvent.

6.23.3.10 virtual MamdaFieldState Wombat::MamdaConcreteBasicEvent::getSymbolFieldState () const
[virtual]

Get the string symbol field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.23.3.11 virtual MamdaFieldState Wombat::MamdaConcreteBasicEvent::getPartIdFieldState () const
[virtual]

Get the participant identifier field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.23.3.12 virtual MamdaFieldState Wombat::MamdaConcreteBasicEvent::getEventSeqNumFieldState () const
[virtual]

Get the event sequence number field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.23.3.13 virtual MamdaFieldState Wombat::MamdaConcreteBasicEvent::getEventTimeFieldState () const
[virtual]

Get the event time field state.
6.23 Wombat::MamdaConcreteBasicEvent Class Reference

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.23.3.14 virtual MamdaFieldState Wombat::MamdaConcreteBasicEvent::getSrcTimeFieldState () const [virtual]

Get the source time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.23.3.15 virtual MamdaFieldState Wombat::MamdaConcreteBasicEvent::getActivityTimeFieldState () const [virtual]

Get the activity time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.23.3.16 virtual MamdaFieldState Wombat::MamdaConcreteBasicEvent::getLineTimeFieldState () const [virtual]

Get the line time of the update.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.23.3.17 virtual MamdaFieldState Wombat::MamdaConcreteBasicEvent::getSendTimeFieldState () const
    [virtual]

Get the send time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.23.3.18 virtual MamdaFieldState Wombat::MamdaConcreteBasicEvent::getMsgQualFieldState () const
    [virtual]

Get the message qualifier field state.

**Returns:**

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.
6.23 Wombat::MamdaConcreteBasicEvent Class Reference

6.23.3.19 virtual void Wombat::MamdaConcreteBasicEvent::setSymbol (const char * value) [virtual]

6.23.3.20 virtual void Wombat::MamdaConcreteBasicEvent::setPartId (const char * value) [virtual]

6.23.3.21 virtual void Wombat::MamdaConcreteBasicEvent::setEventSeqNum (mama_seqnum_t value) [virtual]

6.23.3.22 virtual void Wombat::MamdaConcreteBasicEvent::setEventTime (const MamaDateTime & value) [virtual]

6.23.3.23 virtual void Wombat::MamdaConcreteBasicEvent::setSrcTime (const MamaDateTime & value) [virtual]

6.23.3.24 virtual void Wombat::MamdaConcreteBasicEvent::setActivityTime (const MamaDateTime & value) [virtual]

6.23.3.25 virtual void Wombat::MamdaConcreteBasicEvent::setLineTime (const MamaDateTime & value) [virtual]

6.23.3.26 virtual void Wombat::MamdaConcreteBasicEvent::setSendTime (const MamaDateTime & value) [virtual]

6.23.3.27 virtual void Wombat::MamdaConcreteBasicEvent::setMsgQual (const MamaMsgQual & value) [virtual]

The documentation for this class was generated from the following file:

- MamdaConcreteBasicEvent.h
6.24 Wombat::MamdaContainsAllQuery Class Reference

#include <MamdaQuery.h>

Inheritance diagram for Wombat::MamdaContainsAllQuery::

```
```

Public Member Functions

- **MamdaContainsAllQuery** (const char *field, const char *item)
- **MamdaContainsAllQuery** (const char *field, double item)
- **MamdaContainsAllQuery** (const char *field, int item)
- **bool addItem** (const char *item)
- **bool addItem** (double item)
- **bool addItem** (int item)
- **bool getXML** (char *result)
- **int getDepth** ()
6.24 Wombat::MamdaContainsAllQuery Class Reference

6.24.1 Constructor & Destructor Documentation

6.24.1.1 Wombat::MamdaContainsAllQuery::MamdaContainsAllQuery (const char *field, const char *item)

6.24.1.2 Wombat::MamdaContainsAllQuery::MamdaContainsAllQuery (const char *field, double item)

6.24.1.3 Wombat::MamdaContainsAllQuery::MamdaContainsAllQuery (const char *field, int item)

6.24.2 Member Function Documentation

6.24.2.1 bool Wombat::MamdaContainsAllQuery::addItem (const char *item)

6.24.2.2 bool Wombat::MamdaContainsAllQuery::addItem (double item)

6.24.2.3 bool Wombat::MamdaContainsAllQuery::addItem (int item)

6.24.2.4 bool Wombat::MamdaContainsAllQuery::getXML (char *result) [virtual]

Implements Wombat::MamdaQuery.

6.24.2.5 int Wombat::MamdaContainsAllQuery::getDepth () [virtual]

Implements Wombat::MamdaQuery.

The documentation for this class was generated from the following file:

- MamdaQuery.h
#include <MamdaQuery.h>

Inheritance diagram for Wombat::MamdaContainsQuery:

```
Wombat::MamdaQuery

Wombat::MamdaContainsQuery
```

## Public Member Functions

- `MamdaContainsQuery` (const char *field, const char *item)
- `MamdaContainsQuery` (const char *field, const double item)
- `MamdaContainsQuery` (const char *field, int item)
- `bool addItem` (const char *item)
- `bool addItem` (double item)
- `bool addItem` (int item)
- `bool addItem` (bool item)
- `bool getXML` (char *result)
- `int getDepth` ()
6.25 Wombat::MamdaContainsQuery Class Reference

6.25.1 Constructor & Destructor Documentation

6.25.1.1 Wombat::MamdaContainsQuery::MamdaContainsQuery (const char ∗field, const char ∗item)

6.25.1.2 Wombat::MamdaContainsQuery::MamdaContainsQuery (const char ∗field, const double item)

6.25.1.3 Wombat::MamdaContainsQuery::MamdaContainsQuery (const char ∗field, int item)

6.25.2 Member Function Documentation

6.25.2.1 bool Wombat::MamdaContainsQuery::addItem (const char ∗item)

6.25.2.2 bool Wombat::MamdaContainsQuery::addItem (double item)

6.25.2.3 bool Wombat::MamdaContainsQuery::addItem (int item)

6.25.2.4 bool Wombat::MamdaContainsQuery::addItem (bool item)

6.25.2.5 bool Wombat::MamdaContainsQuery::getXML (char ∗result) [virtual]

Implements Wombat::MamdaQuery.

6.25.2.6 int Wombat::MamdaContainsQuery::getDepth () [virtual]

Implements Wombat::MamdaQuery.

The documentation for this class was generated from the following file:

- MamdaQuery.h
6.26 Wombat::MamdaCurrencyFields Class Reference

```cpp
#include <MamdaCurrencyFields.h>
```

**Static Public Member Functions**

- static void `setDictionary` (const MamaDictionary &dictionary)
- static void `reset` ()
  
  *Reset the dictionary for fundamental update fields.*

- static uint16_t `getMaxFid` ()
- static bool `isSet` ()

**Static Public Attributes**

- static const MamaFieldDescriptor * `BID_PRICE`
- static const MamaFieldDescriptor * `ASK_PRICE`

### 6.26.1 Member Function Documentation

#### 6.26.1.1 static void Wombat::MamdaCurrencyFields::setDictionary (const MamaDictionary &dictionary) [static]

Reset the dictionary for fundamental update fields.

#### 6.26.1.2 static void Wombat::MamdaCurrencyFields::reset () [static]

Reset the dictionary for fundamental update fields.

#### 6.26.1.3 static uint16_t Wombat::MamdaCurrencyFields::getMaxFid () [static]

#### 6.26.1.4 static bool Wombat::MamdaCurrencyFields::isSet () [static]

### 6.26.2 Member Data Documentation

#### 6.26.2.1 const MamaFieldDescriptor* Wombat::MamdaCurrencyFields::BID_PRICE [static]

#### 6.26.2.2 const MamaFieldDescriptor* Wombat::MamdaCurrencyFields::ASK_PRICE [static]

The documentation for this class was generated from the following file:
- MamdaCurrencyFields.h
6.27 Wombat::MamdaCurrencyHandler Class Reference

MamdaCurrencyHandler is an interface for applications that want to have an easy way to access currency data.

```cpp
#include <MamdaCurrencyHandler.h>
```

Public Member Functions

- virtual void `onCurrencyRecap (MamdaSubscription *subscription, MamdaCurrencyListener &listener, const MamaMsg &msg, const MamdaCurrencyRecap &recap)=0`
  
  Method invoked when the current last-currency information for the security is available.

- virtual void `onCurrencyUpdate (MamdaSubscription *subscription, MamdaCurrencyListener &listener, const MamaMsg &msg, const MamdaCurrencyRecap &recap, const MamdaCurrencyUpdate &update)=0`
  
  Method invoked when one or more of the Currency fields have been updated by one of the following market data events:
  - Initial image.

- virtual `~MamdaCurrencyHandler ()`

6.27.1 Detailed Description

MamdaCurrencyHandler is an interface for applications that want to have an easy way to access currency data.

The interface defines callback methods for receiving updates on currency data.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 virtual Wombat::MamdaCurrencyHandler::~MamdaCurrencyHandler () [virtual]
6.27.3 Member Function Documentation

6.27.3.1 virtual void Wombat::MamdaCurrencyHandler::onCurrencyRecap
(MamdaSubscription * subscription, MamdaCurrencyListener &
listener, const MamaMsg & msg, const MamdaCurrencyRecap &
recap) [pure virtual]

Method invoked when the current last-currency information for the security is available.
The reason for the invocation may be any of the following:

- Initial image.
- Recap update (e.g., after server fault tolerant event or data quality event.)
- After stale status removed.

Parameters:

subscription The subscription which received the update.

listener The listener which invoked this callback.

msg The MamaMsg that triggered this invocation.

recap Access to the full quote recap details.

6.27.3.2 virtual void Wombat::MamdaCurrencyHandler::onCurrencyUpdate
(MamdaSubscription * subscription, MamdaCurrencyListener &
listener, const MamaMsg & msg, const MamdaCurrencyRecap &
recap, const MamdaCurrencyUpdate & update) [pure virtual]

Method invoked when one or more of the Currency fields have been updated by one of the
following market data events:

- Initial image.
- Recap update (e.g., after server fault tolerant event or data quality event.)
- Generic update.

Parameters:

subscription The subscription which received the update.

listener The listener which invoked this callback.

msg The MamaMsg that triggered this invocation.
**recap**  The MamaMsg that triggered this invocation.

**update**  Access to the currency update details.

The documentation for this class was generated from the following file:

- MamdaCurrencyHandler.h
**6.28 Wombat::MamdaCurrencyListener Class Reference**

*MamdaCurrencyListener* is a class that specializes in handling currency data. Developers provide their own implementation of the *MamdaCurrencyHandler* interface and will be delivered notifications for updates in the currency data.

```cpp
#include <MamdaCurrencyListener.h>
```

Inheritance diagram for Wombat::MamdaCurrencyListener:

```
Wombat::MamdaMsgListener Wombat::MamdaCurrencyRecap Wombat::MamdaCurrencyUpdate
```

**Public Member Functions**

- `MamdaCurrencyListener ()`
- `virtual ~MamdaCurrencyListener ()`
- `void addHandler (MamdaCurrencyHandler *handler)`
- `const char * getSymbol () const`
  - *Get the instruments string symbol.*
- `const char * getPartId () const`
  - *Get the participant identifier.*
- `const MamaDateTime & getSrcTime () const`
  - *Get the source time.*
- `const MamaDateTime & getActivityTime () const`
  - *Get the activity time.*
- `const MamaDateTime & getLineTime () const`
  - *Get the line time.*
- `const MamaDateTime & getSendTime () const`
  - *Get the send time.*
- `mama_seqnum_t getEventSeqNum () const`

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
Get the event sequence number.

- const MamaDateTime & getEventTime () const
  Get the event time.

- const MamaMsgQual & getMsgQual () const
  Get the message qualifier.

- bool isInitialised () const

- MamdaFieldState getSymbolFieldState () const
  Get the string symbol field state.

- MamdaFieldState getPartIdFieldState () const
  Get the participant identifier field state.

- MamdaFieldState getSrcTimeFieldState () const
  Get the source time field state.

- MamdaFieldState getActivityTimeFieldState () const
  Get the activity time field state.

- MamdaFieldState getLineTimeFieldState () const
  Get the line time of the update.

- MamdaFieldState getSendTimeFieldState () const
  Get the send time field state.

- MamdaFieldState getEventSeqNumFieldState () const
  Get the event sequence number field state.

- MamdaFieldState getEventTimeFieldState () const
  Get the event time field state.

- MamdaFieldState getMsgQualFieldState () const
  Get the message qualifier field state.

- const MamaPrice & getBidPrice () const
  Get the currency bid price.

- const MamaPrice & getAskPrice () const
  Get the currency ask price.

- MamdaFieldState getBidPriceFieldState () const
Get the quote ask price fieldState.

- MamdaFieldState getAskPriceFieldState () const
  Get the quote ask price fieldState.

- virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)
  Implementation of MamdaListener interface.

6.28.1 Detailed Description

MamdaCurrencyListener is a class that specializes in handling currency data. Developers provide their own implementation of the MamdaCurrencyHandler interface and will be delivered notifications for updates in the currency data.

An obvious application for this MAMDA class is any kind of currency analysis application.

MamdaCurrencyListener should initialize the MamdaCurrencyFields class prior to receiving the first message by calling MamdaCurrencyFields::setDictionary() with a valid dictionary object which contains Currency related fields.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 Wombat::MamdaCurrencyListener::MamdaCurrencyListener ()

6.28.2.2 virtual Wombat::MamdaCurrencyListener::~MamdaCurrencyListener () [virtual]

6.28.3 Member Function Documentation

6.28.3.1 void Wombat::MamdaCurrencyListener::addHandler (MamdaCurrencyHandler *handler)

6.28.3.2 const char* Wombat::MamdaCurrencyListener::getSymbol () const [virtual]

Get the instruments string symbol.

Returns:

Symbol. This is the "well-known" symbol for the security, including any symbology mapping performed by the publisher.
Implements Wombat::MamdaBasicEvent.

6.28.3.3 const char* Wombat::MamdaCurrencyListener::getPartId () const [virtual]

Get the participant identifier.

Returns:
Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).

Implements Wombat::MamdaBasicEvent.

6.28.3.4 const MamaDateTime& Wombat::MamdaCurrencyListener::getSrcTime () const [virtual]

Get the source time.

Returns:
Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.28.3.5 const MamaDateTime& Wombat::MamdaCurrencyListener::getActivityTime () const [virtual]

Get the activity time.

Returns:
Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements Wombat::MamdaBasicEvent.

6.28.3.6 const MamaDateTime& Wombat::MamdaCurrencyListener::getLineTime () const [virtual]

Get the line time.
Returns:

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

6.28.3.7 const MamaDateTime& Wombat::MamdaCurrencyListener::getSendTime () const [virtual]

Get the send time.

Returns:

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime()).

Implements Wombat::MamdaBasicEvent.

6.28.3.8 mama_seqnum_t Wombat::MamdaCurrencyListener::getEventSeqNum () const [virtual]

Get the event sequence number.

Returns:

Source sequence number. The exchange generated sequence number.

Implements Wombat::MamdaBasicEvent.

6.28.3.9 const MamaDateTime& Wombat::MamdaCurrencyListener::getEventTime () const [virtual]

Get the event time.

Returns:

Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.
Implements `Wombat::MamdaBasicEvent`.

### 6.28.3.10 const MamaMsgQual& Wombat::MamdaCurrencyListener::getMsgQual () const

Get the message qualifier.

**Returns:**

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements `Wombat::MamdaBasicEvent`.

### 6.28.3.11 bool Wombat::MamdaCurrencyListener::isInitialised () const

### 6.28.3.12 MamdaFieldState Wombat::MamdaCurrencyListener::getSymbolFieldState () const

Get the string symbol field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements `Wombat::MamdaBasicEvent`.

### 6.28.3.13 MamdaFieldState Wombat::MamdaCurrencyListener::getPartIdFieldState () const

Get the participant identifier field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements `Wombat::MamdaBasicEvent`.

### 6.28.3.14 MamdaFieldState Wombat::MamdaCurrencyListener::getSrcTimeFieldState () const

Get the source time field state.
Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.28.3.15 MamdaFieldState Wombat::MamdaCurrencyListener::getActivityTimeFieldState () const [virtual]

Get the activity time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.28.3.16 MamdaFieldState Wombat::MamdaCurrencyListener::getLineTimeFieldState () const [virtual]

Get the line time of the update.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.28.3.17 MamdaFieldState Wombat::MamdaCurrencyListener::getSendTimeFieldState () const [virtual]

Get the send time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.28.3.18 MamdaFieldState Wombat::MamdaCurrencyListener::getEventSeqNumFieldState () const [virtual]

Get the event sequence number field state.
Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.28.3.19 MamdaFieldState Wombat::MamdaCurrencyListener::getEvent-
TimeFieldState () const [virtual]

Get the event time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.28.3.20 MamdaFieldState Wombat::MamdaCurrencyListener::getMsgQual-
FieldState () const [virtual]

Get the message qualifier field state.

Returns:

Message Qualifier. The message qualifier provides information in relation to mes-
sages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.28.3.21 const MamaPrice& Wombat::MamdaCurrencyListener::getBidPrice
() const [virtual]

Get the currency bid price.

Returns:

Bid price. The highest price that the representative party/group is willing to pay to
buy the security.

Implements Wombat::MamdaCurrencyRecap.

6.28.3.22 const MamaPrice& Wombat::MamdaCurrencyListener::getAskPrice
() const [virtual]

Get the currency ask price.
Returns:
Ask price. The lowest price that the representative party/group is willing to take to sell the security.

Implements Wombat::MamdaCurrencyRecap.

6.28.3.23 MamdaFieldState Wombat::MamdaCurrencyListener::getBidPriceFieldState () const [virtual]

Get the quote ask price fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaCurrencyRecap.

6.28.3.24 MamdaFieldState Wombat::MamdaCurrencyListener::getAskPriceFieldState () const [virtual]

Get the quote ask price fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaCurrencyRecap.

6.28.3.25 virtual void Wombat::MamdaCurrencyListener::onMsg (MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [virtual]

Implementation of MamdaListener interface.

Exceptions:

<\textit{MamaStatus}> [MAMDA default implementation.]

Implements Wombat::MamdaMsgListener.

The documentation for this class was generated from the following file:

- MamdaCurrencyListener.h
6.29 Wombat::MamdaCurrencyRecap Class Reference

MamdaCurrencyRecap is an interface that provides access to the currency related fields.

#include <MamdaCurrencyRecap.h>

Inheritance diagram for Wombat::MamdaCurrencyRecap::

Wombat::MamdaBasicRecap

Wombat::MamdaCurrencyRecap

Wombat::MamdaCurrencyListener

Public Member Functions

• virtual const MamaPrice & getBidPrice () const =0
  
  Get the currency bid price.

• virtual const MamaPrice & getAskPrice () const =0
  
  Get the currency ask price.

• virtual MamdaFieldState getBidPriceFieldState () const =0
  
  Get the quote ask price fieldState.

• virtual MamdaFieldState getAskPriceFieldState () const =0
  
  Get the quote ask price fieldState.

• virtual ~MamdaCurrencyRecap ()

6.29.1 Detailed Description

MamdaCurrencyRecap is an interface that provides access to the currency related fields.
6.29 Wombat::MamdaCurrencyRecap Class Reference

6.29.2 Constructor & Destructor Documentation

6.29.2.1 virtual Wombat::MamdaCurrencyRecap::~MamdaCurrencyRecap ()
[virtual]

70 {};

6.29.3 Member Function Documentation

6.29.3.1 virtual const MamaPrice& Wombat::MamdaCurrencyRecap::getBidPrice () const
[pure virtual]

Get the currency bid price.

Returns:
Bid price. The highest price that the representative party/group is willing to pay to
buy the security.

Implemented in Wombat::MamdaCurrencyListener.

6.29.3.2 virtual const MamaPrice& Wombat::MamdaCurrencyRecap::getAskPrice () const
[pure virtual]

Get the currency ask price.

Returns:
Ask price. The lowest price that the representative party/group is willing to take to
sell the security.

Implemented in Wombat::MamdaCurrencyListener.

6.29.3.3 virtual MamdaFieldState Wombat::MamdaCurrencyRecap::getBidPriceFieldState () const
[pure virtual]

Get the quote ask price fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaCurrencyListener.
6.29.3.4 virtual MamdaFieldState Wombat::MamdaCurrency-Recap::getAskPriceFieldState () const [pure virtual]

Get the quote ask price fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaCurrencyListener.

The documentation for this class was generated from the following file:

- MamdaCurrencyRecap.h
MamdaCurrencyUpdate is an interface that provides access to the currency related fields.

#include <MamdaCurrencyUpdate.h>

Inheritance diagram for Wombat::MamdaCurrencyUpdate::

```
Wombat::MamdaBasicEvent

Wombat::MamdaCurrencyUpdate

Wombat::MamdaCurrencyListener
```

Public Member Functions

- virtual const MamaPrice & getBidPrice () const =0
  
  Get the currency bid price.

- virtual const MamaPrice & getAskPrice () const =0
  
  Get the currency ask price.

- virtual MamdaFieldState getBidPriceFieldState () const =0
  
  Get the quote ask price fieldState.

- virtual MamdaFieldState getAskPriceFieldState () const =0
  
  Get the quote ask price fieldState.

- virtual ~MamdaCurrencyUpdate ()

6.30.1 Detailed Description

MamdaCurrencyUpdate is an interface that provides access to the currency related fields.
6.30.2 Constructor & Destructor Documentation

6.30.2.1 virtual Wombat::MamdaCurrencyUpdate::~MamdaCurrencyUpdate () [virtual]

70 {};

6.30.3 Member Function Documentation

6.30.3.1 virtual const MamaPrice& Wombat::MamdaCurrencyUpdate::getBidPrice () const [pure]
virtual]

Get the currency bid price.

Returns:

Bid price. The highest price that the representative party/group is willing to pay to buy the security.

Implemented in Wombat::MamdaCurrencyListener.

6.30.3.2 virtual const MamaPrice& Wombat::MamdaCurrencyUpdate::getAskPrice () const [pure]
virtual]

Get the currency ask price.

Returns:

Ask price. The lowest price that the representative party/group is willing to take to sell the security.

Implemented in Wombat::MamdaCurrencyListener.

6.30.3.3 virtual MamdaFieldState Wombat::MamdaCurrencyUpdate::getBidPriceFieldState () const [pure]
virtual]

Get the quote ask price fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaCurrencyListener.
virtual MamdaFieldState Wombat::MamdaCurrencyUpdate::getAskPriceFieldState () const [pure virtual]

Get the quote ask price fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaCurrencyListener.

The documentation for this class was generated from the following file:

- MamdaCurrencyUpdate.h
6.31 Wombat::MamdaDataException Class Reference

MAMDA data exceptions.

```cpp
#include <MamdaDataException.h>
```

Public Member Functions

- **MamdaDataException (const string &message)**
  
  Constructs a new exception with the specified detail message.

6.31.1 Detailed Description

MAMDA data exceptions.

6.31.2 Constructor & Destructor Documentation

6.31.2.1 Wombat::MamdaDataException::MamdaDataException (const string & message)

Constructs a new exception with the specified detail message.

```cpp
45    : invalid_argument (message)
46    {
47    }
```

The documentation for this class was generated from the following file:

- **MamdaDataException.h**
6.32 Wombat::MamdaDateQuery Class Reference

#include <MamdaQuery.h>

Inheritance diagram for Wombat::MamdaDateQuery::

```
Wombat::MamdaQuery
<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
</table>
| Wombat::MamdaDateQuery
```

Public Member Functions

- MamdaDateQuery (MamaDateTime &start, MamaDateTime &end)
- bool getXML (char *result)
- int getDepth ()

6.32.1 Constructor & Destructor Documentation

6.32.1.1 Wombat::MamdaDateQuery::MamdaDateQuery (MamaDateTime & start, MamaDateTime & end)

6.32.2 Member Function Documentation

6.32.2.1 bool Wombat::MamdaDateQuery::getXML (char * result) [virtual]

Implements Wombat::MamdaQuery.

6.32.2.2 int Wombat::MamdaDateQuery::getDepth () [virtual]

Implements Wombat::MamdaQuery.

The documentation for this class was generated from the following file:

- MamdaQuery.h

---

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.33 Wombat::MamdaEqualsQuery Class Reference

#include <MamdaQuery.h>

Inheritance diagram for Wombat::MamdaEqualsQuery:

Wombat::MamdaEqualsQuery
Wombat::MamdaQuery

Public Member Functions

- MamdaEqualsQuery (const char *field, const char *val)
- MamdaEqualsQuery (const char *field, double val)
- MamdaEqualsQuery (const char *field, int val)
- MamdaEqualsQuery (const char *field, bool val)
- bool addItem (const char *item)
- bool addItem (double item)
- bool addItem (int item)
- bool addItem (bool item)
- bool getXML (char *result)
- int getDepth ()

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.33.1 Constructor & Destructor Documentation

6.33.1.1 Wombat::MamdaEqualsQuery::MamdaEqualsQuery (const char * field, const char * val)

6.33.1.2 Wombat::MamdaEqualsQuery::MamdaEqualsQuery (const char * field, double val)

6.33.1.3 Wombat::MamdaEqualsQuery::MamdaEqualsQuery (const char * field, int val)

6.33.1.4 Wombat::MamdaEqualsQuery::MamdaEqualsQuery (const char * field, bool val)

6.33.2 Member Function Documentation

6.33.2.1 bool Wombat::MamdaEqualsQuery::addItem (const char * item)

6.33.2.2 bool Wombat::MamdaEqualsQuery::addItem (double item)

6.33.2.3 bool Wombat::MamdaEqualsQuery::addItem (int item)

6.33.2.4 bool Wombat::MamdaEqualsQuery::addItem (bool item)

6.33.2.5 bool Wombat::MamdaEqualsQuery::getXML (char * result) [virtual]

Implements Wombat::MamdaQuery.

6.33.2.6 int Wombat::MamdaEqualsQuery::getDepth () [virtual]

Implements Wombat::MamdaQuery.

The documentation for this class was generated from the following file:

- MamdaQuery.h
6.34 Wombat::MamdaErrorListener Class Reference

MamdaErrorListener defines an interface for handling error notifications for a Mamda-Subscription.

#include <MamdaErrorListener.h>

Public Member Functions

• virtual void onError (MamdaSubscription *subscription, MamdaErrorSeverity severity, MamdaErrorCode errorCode, const char *errorStr)=0
  Provide a callback to handle errors.

• virtual ~MamdaErrorListener ()

6.34.1 Detailed Description

MamdaErrorListener defines an interface for handling error notifications for a Mamda-Subscription.

6.34.2 Constructor & Destructor Documentation

6.34.2.1 virtual Wombat::MamdaErrorListener::~MamdaErrorListener ()
  [virtual]

79 {};

6.34.3 Member Function Documentation

6.34.3.1 virtual void Wombat::MamdaErrorListener::onError
  (MamdaSubscription *subscription, MamdaErrorSeverity severity, MamdaErrorCode errorCode, const char *errorStr) [pure
  virtual]

Provide a callback to handle errors.
The severity is intended as a hint to indicate whether the error is recoverable.
The documentation for this class was generated from the following file:

• MamdaErrorListener.h
The documentation for this class was generated from the following file:

- `MamdaFields.h`
6.36 Wombat::MamdaFundamentalFields Class Reference

#include <MamdaFundamentalFields.h>

Static Public Member Functions

• static void setDictionary (const MamaDictionary &dictionary)
• static void reset ()
  Reset the dictionary for fundamental update fields.
• static uint16_t getMaxFid ()
• static bool isSet ()

Static Public Attributes

• static const MamaFieldDescriptor * CORP_ACT_TYPE
• static const MamaFieldDescriptor * DIVIDEND_PRICE
• static const MamaFieldDescriptor * DIVIDEND_FREQ
• static const char * DIV_FREQ_NONE
• static const char * DIV_FREQ_MONTHLY
• static const char * DIV_FREQ_QUARTERLY
• static const char * DIV_FREQ_SEMI_ANNUALLY
• static const char * DIV_FREQ_ANNUALLY
• static const char * DIV_FREQ_SPECIAL
• static const char * DIV_FREQ_IRREGULAR
• static const char * DIV_FREQ_INVALID
• static const MamaFieldDescriptor * DIVIDEND_EX_DATE
• static const MamaFieldDescriptor * DIVIDEND_PAY_DATE
• static const MamaFieldDescriptor * DIVIDEND_REC_DATE
• static const MamaFieldDescriptor * DIVIDEND_CURRENCY
• static const MamaFieldDescriptor * SHARES_OUT
• static const MamaFieldDescriptor * SHARES_FLOAT
• static const MamaFieldDescriptor * SHARES_AUTH
• static const MamaFieldDescriptor * EARN_PER_SHARE
• static const MamaFieldDescriptor * VOLATILITY
• static const MamaFieldDescriptor * PRICE_EARN_RATIO
• static const MamaFieldDescriptor * YIELD
• static const MamaFieldDescriptor * MRKT_SEGM_NATIVE
• static const MamaFieldDescriptor * MRKT_SECT_NATIVE
• static const MamaFieldDescriptor * MRKT_SEGMENT
6.36 Wombat::MamdaFundamentalFields Class Reference

- static const char * MRKT_SEGMENT_NONE

- static const char * MRKT_SEGMENT_EMPTY

- static const MamaFieldDescriptor * MRKT_SECTOR

- static const char * MRKT_SECTOR_NONE

- static const char * MRKT_SECTOR_EMPTY

- static const MamaFieldDescriptor * RISK_FREE_RATE

- static const MamaFieldDescriptor * HIST_VOLATILITY

6.36.1 Member Function Documentation

6.36.1.1 static void Wombat::MamdaFundamentalFields::setDictionary (const MamaDictionary & dictionary) [static]

6.36.1.2 static void Wombat::MamdaFundamentalFields::reset () [static]

Reset the dictionary for fundamental update fields.
6.36.1.3 static uint16_t Wombat::MamdaFundamentalFields::getMaxFid ()
[static]

6.36.1.4 static bool Wombat::MamdaFundamentalFields::isSet () [static]

6.36.2 Member Data Documentation

6.36.2.1 const MamaFieldDescriptor* Wombat::MamdaFundamentalFields::CORP_ACT_TYPE [static]

6.36.2.2 const MamaFieldDescriptor* Wombat::MamdaFundamentalFields::DIVIDEND_PRICE [static]

6.36.2.3 const MamaFieldDescriptor* Wombat::MamdaFundamentalFields::DIVIDEND_FREQ [static]

6.36.2.4 const char*  Wombat::MamdaFundamentalFields::DIV_FREQ_NONE [static]

6.36.2.5 const char*  Wombat::MamdaFundamentalFields::DIV_FREQ_MONTHLY [static]

6.36.2.6 const char*  Wombat::MamdaFundamentalFields::DIV_FREQ_QUARTERLY [static]

6.36.2.7 const char*  Wombat::MamdaFundamentalFields::DIV_FREQ_SEMI_ANNUALLY [static]

6.36.2.8 const char*  Wombat::MamdaFundamentalFields::DIV_FREQ_ANNUALLY [static]

6.36.2.9 const char*  Wombat::MamdaFundamentalFields::DIV_FREQ_SPECIAL [static]

6.36.2.10 const char* Wombat::MamdaFundamentalFields::DIV_FREQ_IRREGULAR [static]

6.36.2.11 const char* Wombat::MamdaFundamentalFields::DIV_FREQ_INVALID [static]

6.36.2.12 const MamaFieldDescriptor* Wombat::MamdaFundamentalFields::DIVIDEND_EX_DATE [static]

6.36.2.13 const MamaFieldDescriptor* Wombat::MamdaFundamentalFields::DIVIDEND_PAY_DATE [static]

6.36.2.14 const MamaFieldDescriptor* Wombat::MamdaFundamentalFields::DIVIDEND_REC_DATE [static]

6.36.2.15 const MamaFieldDescriptor* Wombat::MamdaFundamentalFields::DIVIDEND_CURRENCY [static]
• MamdaFundamentalFields.h
MamdaFundamentalHandler is an interface for applications that want to have an easy way to access fundamental equity pricing/analysis attributes, indicators and ratios.

#include <MamdaFundamentalHandler.h>

Public Member Functions

- virtual void onFundamentals (MamdaSubscription *subscription, MamdaFundamentalListener &listener, const MamaMsg &msg, const MamdaFundamentals &fundas)=0
  
  Method invoked when one or more of the Fundamental fields have been updated by one of the following market data events:
  - Initial image.

- virtual ~MamdaFundamentalHandler ()

6.37.1 Detailed Description

MamdaFundamentalHandler is an interface for applications that want to have an easy way to access fundamental equity pricing/analysis attributes, indicators and ratios.

The interface defines a single callback method for receiving updates on fundamental data.

6.37.2 Constructor & Destructor Documentation

6.37.2.1 virtual Wombat::MamdaFundamentalHandler::~MamdaFundamentalHandler () [virtual]

61 {};

6.37.3 Member Function Documentation

6.37.3.1 virtual void Wombat::MamdaFundamentalHandler::onFundamentals (MamdaSubscription *subscription, MamdaFundamentalListener &listener, const MamaMsg &msg, const MamdaFundamentals &fundas) [pure virtual]

Method invoked when one or more of the Fundamental fields have been updated by one of the following market data events:
• Initial image.

• Recap update (e.g., after server fault tolerant event or data quality event.)

• Generic update.

**Parameters:**

- `subscription`  The subscription which received the update.
- `listener`  The listener which invoked this callback.
- `msg`  The MamaMsg that triggered this invocation.
- `fundas`  Access to the fundamental update details.

The documentation for this class was generated from the following file:

- `MamdaFundamentalHandler.h`
6.38 Wombat::MamdaFundamentalListener Class Reference

MamdaFundamentalListener is a class that specializes in handling fundamental equity pricing/analysis attributes, indicators and ratios.

#include <MamdaFundamentalListener.h>

Inheritance diagram for Wombat::MamdaFundamentalListener:

```
Wombat::MamdaBasicRecap

Wombat::MamdaMsgListener   Wombat::MamdaFundamentals

Wombat::MamdaFundamentalListener
```

Public Member Functions

- MamdaFundamentalListener ()
- virtual ~MamdaFundamentalListener ()
- void addHandler (MamdaFundamentalHandler *handler)
- const char * getSymbol () const
  
  Get the string symbol for the instrument.

- const char * getPartId () const
  
  Get the participant identifier.

- const MamaDateTime & getSrcTime () const
  
  Get the source time of the update.

- const MamaDateTime & getActivityTime () const
  
  Get the activity time of the update.

- const MamaDateTime & getLineTime () const
  
  Get the line time of the update.

- const MamaDateTime & getSendTime () const
  
  Get the send time of the update.

- MamdaFieldState getSymbolFieldState () const
Get the string symbol field state for the instrument.

- `MamdaFieldState getPartIdFieldState () const`
  
  Get the participant identifier field state.

- `MamdaFieldState getSrcTimeFieldState () const`
  
  Get the source time field state.

- `MamdaFieldState getActivityTimeFieldState () const`
  
  Get the activity time field state.

- `MamdaFieldState getLineTimeFieldState () const`
  
  Get the line time field state.

- `MamdaFieldState getSendTimeFieldState () const`
  
  Get the send time field state.

- `const char * getCorporateActionType () const`
- `double getDividendPrice () const`
- `const char * getDividendFrequency () const`
- `const char * getDividendExDate () const`
- `const char * getDividendPayDate () const`
- `const char * getDividendRecordDate () const`
- `const char * getDividendCurrency () const`
- `long getSharesOut () const`
- `long getSharesFloat () const`
- `long getSharesAuthorized () const`
- `double getEarningsPerShare () const`
- `double getVolatility () const`
- `double getPriceEarningsRatio () const`
- `double getYield () const`
- `const char * getMarketSegmentNative () const`
- `const char * getMarketSectorNative () const`
- `const char * getMarketSegment () const`
- `const char * getMarketSector () const`
- `double getHistoricalVolatility () const`
- `double getRiskFreeRate () const`
- `MamdaFieldState getCorporateActionTypeFieldState () const`
- `MamdaFieldState getDividendPriceFieldState () const`
- `MamdaFieldState getDividendFrequencyFieldState () const`
- `MamdaFieldState getDividendExDateFieldState () const`
- `MamdaFieldState getDividendPayDateFieldState () const`
- `MamdaFieldState getDividendRecordDateFieldState () const`
• MamdaFieldState getDividendCurrencyFieldState () const
• MamdaFieldState getSharesOutFieldState () const
• MamdaFieldState getSharesFloatFieldState () const
• MamdaFieldState getSharesAuthorizedFieldState () const
• MamdaFieldState getEarningsPerShareFieldState () const
• MamdaFieldState getVolatilityFieldState () const
• MamdaFieldState getPriceEarningsRatioFieldState () const
• MamdaFieldState getYieldFieldState () const
• MamdaFieldState getMarketSegmentNativeFieldState () const
• MamdaFieldState getMarketSectorNativeFieldState () const
• MamdaFieldState getMarketSegmentFieldState () const
• MamdaFieldState getMarketSectorFieldState () const
• MamdaFieldState getHistoricalVolatilityFieldState () const
• MamdaFieldState getRiskFreeRateFieldState () const
• virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)

  Implementation of MamdaListener interface.

6.38.1 Detailed Description

MamdaFundamentalListener is a class that specializes in handling fundamental equity pricing/analysis attributes, indicators and ratios.

Developers provide their own implementation of the MamdaFundamentalHandler interface and will be delivered notifications for updates in the fundamental data. An obvious application for this MAMDA class is any kind of pricing analysis application.

Note: The MamdaFundamentalListener class caches equity pricing/analysis attributes, indicators and ratios. Among other reasons, caching of these fields makes it possible to provide complete fundamental callbacks, even when the publisher (e.g., feed handler) is only publishing deltas containing modified fields.

MamdaFundamentalListener should initialize the MamdaFundamentalFields class prior to receiving the first message by calling MamdaFundamentalFields::setDictionary() with a valid dictionary object which contains Fundamental related fields.
6.38.2 Constructor & Destructor Documentation

6.38.2.1 Wombat::MamdaFundamentalListener::MamdaFundamentalListener ()

6.38.2.2 virtual Wombat::MamdaFundamentalListener::~MamdaFundamentalListener () [virtual]

6.38.3 Member Function Documentation

6.38.3.1 void Wombat::MamdaFundamentalListener::addHandler (MamdaFundamentalHandler * handler)

6.38.3.2 const char * Wombat::MamdaFundamentalListener::getSymbol () const [virtual]

Get the string symbol for the instrument.

Returns:

Symbol. This is the "well-known" symbol for the security, including any symbology mapping performed by the publisher.

Implements Wombat::MamdaBasicRecap.

6.38.3.3 const char * Wombat::MamdaFundamentalListener::getPartId () const [virtual]

Get the participant identifier.

Returns:

Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).

Implements Wombat::MamdaBasicRecap.

6.38.3.4 const MamaDateTime& Wombat::MamdaFundamentalListener::getSourceTime () const [virtual]

Get the source time of the update.

Returns:

Source time. Typically, the exchange generated feed time stamp. This is often
the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicRecap.

6.38.3.5 const MamaDateTime& Wombat::MamdaFundamentalListener::getActivityTime () const [virtual]

Get the activity time of the update.

Returns:

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements Wombat::MamdaBasicRecap.

6.38.3.6 const MamaDateTime& Wombat::MamdaFundamentalListener::getLineTime () const [virtual]

Get the line time of the update.

Returns:

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicRecap.

6.38.3.7 const MamaDateTime& Wombat::MamdaFundamentalListener::getSendTime () const [virtual]

Get the send time of the update.

Returns:

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime().
Implements Wombat::MamdaBasicRecap.

6.38.3.8 MamdaFieldState Wombat::MamdaFundamentalListener::getSymbolFieldState () const [virtual]

Get the string symbol field state for the instrument.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicRecap.

6.38.3.9 MamdaFieldState Wombat::MamdaFundamentalListener::getPartIdFieldState () const [virtual]

Get the participant identifier field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicRecap.

6.38.3.10 MamdaFieldState Wombat::MamdaFundamentalListener::getSrcTimeFieldState () const [virtual]

Get the source time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicRecap.

6.38.3.11 MamdaFieldState Wombat::MamdaFundamentalListener::getActivityTimeFieldState () const [virtual]

Get the activity time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicRecap.
6.38 Wombat::MamdaFundamentalListener Class Reference

6.38.3.12  **MamdaFieldState** Wombat::MamdaFundamentalListener::getLineTimeFieldState () const  [virtual]

Get the line time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaBasicRecap**.

6.38.3.13  **MamdaFieldState** Wombat::MamdaFundamentalListener::getSendTimeFieldState () const  [virtual]

Get the send time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaBasicRecap**.

6.38.3.14  **const char** ∗ Wombat::MamdaFundamentalListener::getCorporateActionType () const  [virtual]

Implements **Wombat::MamdaFundamentals**.

6.38.3.15  **double** Wombat::MamdaFundamentalListener::getDividendPrice () const  [virtual]

Implements **Wombat::MamdaFundamentals**.

6.38.3.16  **const char** ∗ Wombat::MamdaFundamentalListener::getDividendFrequency () const  [virtual]

Implements **Wombat::MamdaFundamentals**.

6.38.3.17  **const char** ∗ Wombat::MamdaFundamentalListener::getDividendExDate () const  [virtual]

Implements **Wombat::MamdaFundamentals**.
6.38.3.18 const char* Wombat::MamdaFundamentalListener::getDividendPayDate () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.19 const char* Wombat::MamdaFundamentalListener::getDividendRecordDate () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.20 const char* Wombat::MamdaFundamentalListener::getDividendCurrency () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.21 long Wombat::MamdaFundamentalListener::getSharesOut () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.22 long Wombat::MamdaFundamentalListener::getSharesFloat () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.23 long Wombat::MamdaFundamentalListener::getSharesAuthorized () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.24 double Wombat::MamdaFundamentalListener::getEarningsPerShare () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.25 double Wombat::MamdaFundamentalListener::getVolatility () const [virtual]

Implements Wombat::MamdaFundamentals.
6.38 Wombat::MamdaFundamentalListener Class Reference

6.38.3.26  double Wombat::MamdaFundamentalListener::getPriceEarnings-Ratio () const  [virtual]
Implements Wombat::MamdaFundamentals.

6.38.3.27  double Wombat::MamdaFundamentalListener::getYield () const  [virtual]
Implements Wombat::MamdaFundamentals.

6.38.3.28  const char ∗ Wombat::MamdaFundamentalListener::getMarket- SegmentNative () const  [virtual]
Implements Wombat::MamdaFundamentals.

6.38.3.29  const char ∗ Wombat::MamdaFundamentalListener::getMarket- SectorNative () const  [virtual]
Implements Wombat::MamdaFundamentals.

6.38.3.30  const char ∗ Wombat::MamdaFundamentalListener::getMarket- Segment () const  [virtual]
Implements Wombat::MamdaFundamentals.

6.38.3.31  const char ∗ Wombat::MamdaFundamentalListener::getMarket- Sector () const  [virtual]
Implements Wombat::MamdaFundamentals.

6.38.3.32  double Wombat::MamdaFundamentalListener::getHistorical- Volatility () const  [virtual]
Implements Wombat::MamdaFundamentals.

6.38.3.33  double Wombat::MamdaFundamentalListener::getRiskFreeRate () const  [virtual]
Implements Wombat::MamdaFundamentals.
6.38.3.34 MamdaFieldState Wombat::MamdaFundamentalListener::getCorporateActionTypeFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.35 MamdaFieldState Wombat::MamdaFundamentalListener::getDividendPriceFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.36 MamdaFieldState Wombat::MamdaFundamentalListener::getDividendFrequencyFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.37 MamdaFieldState Wombat::MamdaFundamentalListener::getDividendExDateFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.38 MamdaFieldState Wombat::MamdaFundamentalListener::getDividendPayDateFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.39 MamdaFieldState Wombat::MamdaFundamentalListener::getDividendRecordDateFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.40 MamdaFieldState Wombat::MamdaFundamentalListener::getDividendCurrencyFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.
6.38 Wombat::MamdaFundamentalListener Class Reference

6.38.3.41 **MamdaFieldState** Wombat::MamdaFundamentalListener::getSharesOutFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.42 **MamdaFieldState** Wombat::MamdaFundamentalListener::getSharesFloatFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.43 **MamdaFieldState** Wombat::MamdaFundamentalListener::getSharesAuthorizedFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.44 **MamdaFieldState** Wombat::MamdaFundamentalListener::getEarningsPerShareFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.45 **MamdaFieldState** Wombat::MamdaFundamentalListener::getVolatilityFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.46 **MamdaFieldState** Wombat::MamdaFundamentalListener::getPriceEarningsRatioFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.

6.38.3.47 **MamdaFieldState** Wombat::MamdaFundamentalListener::getYieldFieldState () const [virtual]

Implements Wombat::MamdaFundamentals.
6.38.3.48  
```
MamdaFieldState Wombat::MamdaFundamentalListener::getMarketSegmentNativeFieldState () const
[virtual]
```

Implements Wombat::MamdaFundamentals.

6.38.3.49  
```
MamdaFieldState Wombat::MamdaFundamentalListener::getMarketSectorNativeFieldState () const
[virtual]
```

Implements Wombat::MamdaFundamentals.

6.38.3.50  
```
MamdaFieldState Wombat::MamdaFundamentalListener::getMarketSegmentFieldState () const
[virtual]
```

Implements Wombat::MamdaFundamentals.

6.38.3.51  
```
MamdaFieldState Wombat::MamdaFundamentalListener::getMarketSectorFieldState () const
[virtual]
```

Implements Wombat::MamdaFundamentals.

6.38.3.52  
```
MamdaFieldState Wombat::MamdaFundamentalListener::getHistoricalVolatilityFieldState () const
[virtual]
```

Implements Wombat::MamdaFundamentals.

6.38.3.53  
```
MamdaFieldState Wombat::MamdaFundamentalListener::getRiskFreeRateFieldState () const
[virtual]
```

Implements Wombat::MamdaFundamentals.

6.38.3.54  
```
virtual void Wombat::MamdaFundamentalListener::onMsg (MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [virtual]
```

Implementation of MamdaListener interface.
Exceptions:

\[ \text{MamaStatus} \]  {NYSE Technologies default implementation.}

Implements \texttt{Wombat::MamdaMsgListener}.

The documentation for this class was generated from the following file:

- \texttt{MamdaFundamentalListener.h}
6.39 Wombat::MamdaFundamentals Class Reference

MamdaFundamentals is an interface that provides access to the fundamental equity pricing/analysis attributes, indicators and ratios.

#include <MamdaFundamentals.h>

Inheritance diagram for Wombat::MamdaFundamentals::

```
Wombat::MamdaBasicRecap

Wombat::MamdaFundamentals

Wombat::MamdaFundamentalListener
```

Public Member Functions

- virtual const char * getCorporateActionType () const =0
- virtual double getDividendPrice () const =0
- virtual const char * getDividendFrequency () const =0
- virtual const char * getDividendExDate () const =0
- virtual const char * getDividendPayDate () const =0
- virtual const char * getDividendRecordDate () const =0
- virtual const char * getDividendCurrency () const =0
- virtual long getSharesOut () const =0
- virtual long getSharesFloat () const =0
- virtual long getSharesAuthorized () const =0
- virtual double getEarningsPerShare () const =0
- virtual double getVolatility () const =0
- virtual double getPriceEarningsRatio () const =0
- virtual double getRiskFreeRate () const =0
- virtual MamdaFieldState getCorporateActionTypeFieldState () const =0
- virtual MamdaFieldState getDividendPriceFieldState () const =0
- virtual MamdaFieldState getDividendFrequencyFieldState () const =0
- virtual MamdaFieldState getDividendExDateFieldState () const =0
6.39 Wombat::MamdaFundamentals Class Reference

- virtual MamdaFieldState getDividendPayDateFieldState () const =0
- virtual MamdaFieldState getDividendRecordDateFieldState () const =0
- virtual MamdaFieldState getDividendCurrencyFieldState () const =0
- virtual MamdaFieldState getSharesOutFieldState () const =0
- virtual MamdaFieldState getSharesFloatFieldState () const =0
- virtual MamdaFieldState getSharesAuthorizedFieldState () const =0
- virtual MamdaFieldState getEarningsPerShareFieldState () const =0
- virtual MamdaFieldState getVolatilityFieldState () const =0
- virtual MamdaFieldState getPriceEarningsRatioFieldState () const =0
- virtual MamdaFieldState getYieldFieldState () const =0
- virtual MamdaFieldState getMarketSegmentNativeFieldState () const =0
- virtual MamdaFieldState getMarketSectorNativeFieldState () const =0
- virtual MamdaFieldState getMarketSegmentFieldState () const =0
- virtual MamdaFieldState getMarketSectorFieldState () const =0
- virtual MamdaFieldState getHistoricalVolatilityFieldState () const =0
- virtual MamdaFieldState getRiskFreeRateFieldState () const =0
- virtual ~MamdaFundamentals ()

6.39.1 Detailed Description

MamdaFundamentals is an interface that provides access to the fundamental equity pricing/analysis attributes, indicators and ratios.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 virtual Wombat::MamdaFundamentals::~MamdaFundamentals ()
    [virtual]

82 {};

6.39.3 Member Function Documentation

6.39.3.1 virtual const char* Wombat::MamdaFundamentals::getCorporateActionType () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.2 virtual double Wombat::MamdaFundamentals::getDividendPrice ()
    const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.
6.39.3.3 virtual const char* Wombat::MamdaFundamentals::getDividendFrequency () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.4 virtual const char* Wombat::MamdaFundamentals::getDividendExDate () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.5 virtual const char* Wombat::MamdaFundamentals::getDividendPayDate () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.6 virtual const char* Wombat::MamdaFundamentals::getDividendRecordDate () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.7 virtual const char* Wombat::MamdaFundamentals::getDividendCurrency () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.8 virtual long Wombat::MamdaFundamentals::getSharesOut () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.9 virtual long Wombat::MamdaFundamentals::getSharesFloat () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.10 virtual long Wombat::MamdaFundamentals::getSharesAuthorized () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.
6.39 Wombat::MamdaFundamentals Class Reference

6.39.3.11 virtual double Wombat::MamdaFundamentals::getEarningsPerShare () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.12 virtual double Wombat::MamdaFundamentals::getVolatility () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.13 virtual double Wombat::MamdaFundamentals::getPriceEarningsRatio () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.14 virtual double Wombat::MamdaFundamentals::getYield () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.15 virtual const char* Wombat::MamdaFundamentals::getMarketSegmentNative () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.16 virtual const char* Wombat::MamdaFundamentals::getMarketSectorNative () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.17 virtual const char* Wombat::MamdaFundamentals::getMarketSegment () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.18 virtual const char* Wombat::MamdaFundamentals::getMarketSector () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.
6.39.3.19 virtual double Wombat::MamdaFundamentals::getHistoricalVolatility () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.20 virtual double Wombat::MamdaFundamentals::getRiskFreeRate () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.21 virtual MamdaFieldState Wombat::MamdaFundamentals::getCorporateActionTypeFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.22 virtual MamdaFieldState Wombat::MamdaFundamentals::getDividendPriceFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.23 virtual MamdaFieldState Wombat::MamdaFundamentals::getDividendFrequencyFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.24 virtual MamdaFieldState Wombat::MamdaFundamentals::getDividendExDateFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.25 virtual MamdaFieldState Wombat::MamdaFundamentals::getDividendPayDateFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.
6.39 Wombat::MamdaFundamentals Class Reference

6.39.3.26 virtual MamdaFieldState Wombat::MamdaFundamentals::getDividendRecordDateFieldState () const [pure
virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.27 virtual MamdaFieldState Wombat::MamdaFundamentals::getDividendCurrencyFieldState () const [pure
virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.28 virtual MamdaFieldState Wombat::MamdaFundamentals::getSharesOutFieldState () const [pure
virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.29 virtual MamdaFieldState Wombat::MamdaFundamentals::getSharesFloatFieldState () const [pure
virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.30 virtual MamdaFieldState Wombat::MamdaFundamentals::getSharesAuthorizedFieldState () const [pure
virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.31 virtual MamdaFieldState Wombat::MamdaFundamentals::getEarningsPerShareFieldState () const [pure
virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.32 virtual MamdaFieldState Wombat::MamdaFundamentals::getVolatilityFieldState () const [pure
virtual]

Implemented in Wombat::MamdaFundamentalListener.
6.39.3.33 virtual MamdaFieldState Wombat::MamdaFundamentals::getPriceEarningsRatioFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.34 virtual MamdaFieldState Wombat::MamdaFundamentals::getYieldFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.35 virtual MamdaFieldState Wombat::MamdaFundamentals::getMarketSegmentNativeFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.36 virtual MamdaFieldState Wombat::MamdaFundamentals::getMarketSectorNativeFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.37 virtual MamdaFieldState Wombat::MamdaFundamentals::getMarketSegmentFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.38 virtual MamdaFieldState Wombat::MamdaFundamentals::getMarketSectorFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

6.39.3.39 virtual MamdaFieldState Wombat::MamdaFundamentals::getHistoricalVolatilityFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.
6.39.3.40 virtual MamdaFieldState Wombat::MamdaFundamentals::getRiskFreeRateFieldState () const [pure virtual]

Implemented in Wombat::MamdaFundamentalListener.

The documentation for this class was generated from the following file:

- MamdaFundamentals.h
#include <MamdaLock.h>

Public Types

- `SHARED`
- `EXCLUSIVE`
- `READ`
- `WRITE`
- `enum Scheme { SHARED, EXCLUSIVE }`
- `enum Type { READ, WRITE }`

Public Member Functions

- `MamdaLock (Scheme scheme, const char *context, int threads=0)`
- `~MamdaLock ()`
- `bool acquire (Type type)`
- `bool release (Type type)`

Static Public Member Functions

- `static const char * type (Type type)`
- `static const char * scheme (Scheme scheme)`
- `static bool lockingEnabled ()`

6.40.1 Member Enumeration Documentation

6.40.1.1 `enum Wombat::MamdaLock::Scheme`

**Enumerator:**

<table>
<thead>
<tr>
<th><code>SHARED</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>EXCLUSIVE</code></td>
</tr>
</tbody>
</table>

```cpp
55 {
56     SHARED,
57     EXCLUSIVE
58 };
```
6.40 Wombat::MamdaLock Class Reference

6.40.1.2 enum Wombat::MamdaLock::Type

Enumerator:

\begin{verbatim}
READ
WRITE
\end{verbatim}

6.40.2 Constructor & Destructor Documentation

6.40.2.1 Wombat::MamdaLock::MamdaLock (Scheme scheme, const char * context, int threads = 0)

6.40.2.2 Wombat::MamdaLock::~MamdaLock ()

6.40.3 Member Function Documentation

6.40.3.1 bool Wombat::MamdaLock::acquire (Type type)

6.40.3.2 bool Wombat::MamdaLock::release (Type type)

6.40.3.3 static const char * Wombat::MamdaLock::type (Type type) [static]

6.40.3.4 static const char * Wombat::MamdaLock::scheme (Scheme scheme) [static]

6.40.3.5 static bool Wombat::MamdaLock::lockingEnabled () [static]

\begin{verbatim}
78 {
79     // This is to be a configuration parameter at some point
80     return true;
81 }
\end{verbatim}

The documentation for this class was generated from the following file:

- MamdaLock.h
6.41 Wombat::MamdaMsgListener Class Reference

MamdaMsgListener defines an interface for handling MAMA messages for a MamdaSubscription.

#include <MamdaMsgListener.h>

Inheritance diagram for Wombat::MamdaMsgListener::

Public Member Functions

- virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)=0
- virtual ~MamdaMsgListener ()

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.41 Wombat::MamdaMsgListener Class Reference

6.41.1 Detailed Description

MamdaMsgListener defines an interface for handling MAMA messages for a MamdaSubscription.

6.41.2 Constructor & Destructor Documentation

6.41.2.1 virtual Wombat::MamdaMsgListener::~MamdaMsgListener ()
[virtual]

45 {};

6.41.3 Member Function Documentation

6.41.3.1 virtual void Wombat::MamdaMsgListener::onMsg
(MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [pure virtual]

Implemented in Wombat::MamdaAuctionListener, Wombat::MamdaCurrencyListener, Wombat::MamdaFundamentalListener, Wombat::MamdaMultiParticipantManager, Wombat::MamdaMultiSecurityManager, Wombat::MamdaOrderImbalanceListener, Wombat::MamdaPubStatusListener, Wombat::MamdaQuoteListener, Wombat::MamdaSecStatusListener, Wombat::MamdaTradeListener, Wombat::MamdaBookAtomicListener, MamdaOrderBookDepthFilter, Wombat::MamdaOrderBookListener, MamdaQuoteToBookListener, and Wombat::MamdaOptionChainListener.

The documentation for this class was generated from the following file:

- MamdaMsgListener.h
The MamdaMultiParticipantHandler class is an interface that allows a developer to be notified dynamically when participants are added to the list.

```cpp
#include <MamdaMultiParticipantHandler.h>
```

### Public Member Functions

- **virtual void onConsolidatedCreate (MamdaSubscription *subscription, MamdaMultiParticipantManager &manager)=0**

  Method invoked when the consolidated trade and BBO quote information for the security has become available.

- **virtual void onParticipantCreate (MamdaSubscription *subscription, MamdaMultiParticipantManager &manager, const char *particpantId, bool isPrimary)=0**

  Method invoked when the trade and quote information for a participant has become available for the security.

- **virtual ~MamdaMultiParticipantHandler ()**

### 6.42.1 Detailed Description

The MamdaMultiParticipantHandler class is an interface that allows a developer to be notified dynamically when participants are added to the list.

This is useful for handling information such as NYSE, AMEX and NASDAQ listed securities (including NQDS). Access to consolidated information (i.e., best bid and offer and consolidated trade info) is also available.

Note that any actions to register per-participant or consolidated listeners can be added up front (and this callback omitted or left empty) if the participant IDs are known beforehand.

### 6.42.2 Constructor & Destructor Documentation

#### 6.42.2.1 virtual Wombat::MamdaMultiParticipantHandler::~MamdaMultiParticipantHandler () [virtual]

```cpp
71 {};
```
6.42.3 Member Function Documentation

6.42.3.1 virtual void Wombat::MamdaMultiParticipantHandler::onConsolidatedCreate (MamdaSubscription * subscription, MamdaMultiParticipantManager & manager) [pure virtual]

Method invoked when the consolidated trade and BBO quote information for the security has become available.

This method is invoked only if there is BBO or consolidated trade information available.

6.42.3.2 virtual void Wombat::MamdaMultiParticipantHandler::onParticipantCreate (MamdaSubscription * subscription, MamdaMultiParticipantManager & manager, const char * participantId, bool isPrimary) [pure virtual]

Method invoked when the trade and quote information for a participant has become available for the security.

This method is invoked only if there is participant quote or trade information available.

isPrimary is not yet supported!

The documentation for this class was generated from the following file:

- MamdaMultiParticipantHandler.h
6.43 Wombat::MamdaMultiParticipantManager
Class Reference

MamdaMultiParticipantManager is a class that manages updates on a consolidated basis for securities that may be traded on multiple exchanges and which may have a national best bid and offer.

```cpp
#include <MamdaMultiParticipantManager.h>
```

Inheritance diagram for Wombat::MamdaMultiParticipantManager:

```
Wombat::MamdaMsgListener
Wombat::MamdaMultiParticipantManager
```

Public Member Functions

- **MamdaMultiParticipantManager** (const char *symbol)
  
  *Create a manager for consolidated securities.*

- **virtual ~MamdaMultiParticipantManager ()**
  
  *Destructor.*

- **void addHandler (MamdaMultiParticipantHandler *handler)**
  
  *Add a specialized handler for notifications about the multi-participant security.*

- **void addConsolidatedListener (MamdaMsgListener *listener)**
  
  *Add a specialized message listener (e.g., a MamdaQuoteListener, MamdaTradeListener, etc.)*

- **void addParticipantListener (MamdaMsgListener *listener, const char *partId)**
  
  *Add a specialized message listener (e.g., a MamdaQuoteListener, MamdaTradeListener, etc.)*

- **void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)**
  
  *Implementation of MamdaMsgListener interface.*
6.43 Wombat::MamdaMultiParticipantManager Class Reference

6.43.1 Detailed Description

MamdaMultiParticipantManager is a class that manages updates on a consolidated basis for securities that may be traded on multiple exchanges and which may have a national best bid and offer.

Developers are notified of each element available for the consolidated security, including the national best bid and offer, and each regional exchange. Developers can pick and choose which elements they wish to provide handling for (e.g., BBO-only, certain regional exchanges, etc.).

Note: the MamdaMultiParticipantManager can also be used for securities that are not traded on multiple exchanges.

6.43.2 Constructor & Destructor Documentation

6.43.2.1 Wombat::MamdaMultiParticipantManager::MamdaMultiParticipantManager (const char ∗ symbol)

Create a manager for consolidated securities.

6.43.2.2 virtual Wombat::MamdaMultiParticipantManager::~MamdaMultiParticipantManager () [virtual]

Destructor.

6.43.3 Member Function Documentation

6.43.3.1 void Wombat::MamdaMultiParticipantManager::addHandler (MamdaMultiParticipantHandler ∗ handler)

Add a specialized handler for notifications about the multi-participant security.

The handler is responsible for initializing any data structures and listeners for each participant as well as the consolidated. Currently, only one handler can be registered.

6.43.3.2 void Wombat::MamdaMultiParticipantManager::addConsolidatedListener (MamdaMsgListener ∗ listener)

Add a specialized message listener (e.g., a MamdaQuoteListener, MamdaTradeListener, etc.) for the consolidated data.
6.43.3.3 void Wombat::MamdaMultiParticipantManager::addParticipantListener (MamdaMsgListener * listener, const char * partId)

Add a specialized message listener (e.g., a MamdaQuoteListener, MamdaTradeListener, etc.) for a participant.

6.43.3.4 void Wombat::MamdaMultiParticipantManager::onMsg (MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [virtual]

Implementation of MamdaMsgListener interface.
Implements Wombat::MamdaMsgListener.
The documentation for this class was generated from the following file:

- MamdaMultiParticipantManager.h
The `MamdaMultiSecurityHandler` class is an interface that allows a developer to be notified dynamically when securities are added to the list.

```
#include <MamdaMultiSecurityHandler.h>
```

### Public Member Functions

- virtual `void onSecurityCreate (MamdaSubscription *subscription, MamdaMultiSecurityManager &manager, const char *securitySymbol, bool isPrimary)=0`

  `Method invoked when the trade and quote information for the security has become available.`

- virtual `~MamdaMultiSecurityHandler ()`

### 6.44.1 Detailed Description

The `MamdaMultiSecurityHandler` class is an interface that allows a developer to be notified dynamically when securities are added to the list.

Access to consolidated information (i.e., best bid and offer and consolidated trade info) is also available.

### 6.44.2 Constructor & Destructor Documentation

#### 6.44.2.1 virtual Wombat::MamdaMultiSecurityHandler::~MamdaMultiSecurityHandler () [virtual]

```
62 {
}
```

### 6.44.3 Member Function Documentation

#### 6.44.3.1 virtual void Wombat::MamdaMultiSecurityHandler::onSecurityCreate (MamdaSubscription * subscription, MamdaMultiSecurityManager & manager, const char * securitySymbol, bool isPrimary) [pure virtual]

Method invoked when the trade and quote information for the security has become available.
Invoked as a result of an initial image on subscription (assuming initial values are enabled) or if an update is received for a security the `MamdaMultiSecurityManager` is unaware of (typically for securities which come online intra day and, for which, we have not received an initial value). This method is invoked for participant and consolidated quote or trade information.

`isPrimary` is not yet supported!

The documentation for this class was generated from the following file:

```
- MamdaMultiSecurityHandler.h
```
MamdaMultiSecurityManager is a class that manages updates on an arbitrary number of securities that may be traded on multiple exchanges.

#include <MamdaMultiSecurityManager.h>

Inheritance diagram for Wombat::MamdaMultiSecurityManager:

```
Wombat::MamdaMsgListener
Wombat::MamdaMultiSecurityManager
```

Public Member Functions

- **MamdaMultiSecurityManager** (const char *symbol)
  
  Create a manager for consolidated securities.

- **virtual ~MamdaMultiSecurityManager ()**
  
  Destructor.

- **void addHandler (MamdaMultiSecurityHandler *handler)**
  
  Add a specialized handler for notifications about each security in the group.

- **void addSecurityListener (MamdaMsgListener *listener, const char *securitySymbol)**
  
  Add a specialized message listener (e.g., a MamdaQuoteListener, MamdaTradeListener, etc.

- **void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)**
  
  Implementation of MamdaMsgListener interface.

6.45.1 Detailed Description

MamdaMultiSecurityManager is a class that manages updates on an arbitrary number of securities that may be traded on multiple exchanges.
Developers are notified of each security available, including the national best bid and offer, and each regional exchange, if applicable. Developers can pick and choose which elements they wish to provide handling for (e.g., BBO-only, certain regional exchanges, etc.).

### 6.45.2 Constructor & Destructor Documentation

#### 6.45.2.1 Wombat::MamdaMultiSecurityManager::MamdaMultiSecurityManager (const char ∗ symbol)

Create a manager for consolidated securities.

#### 6.45.2.2 virtual Wombat::MamdaMultiSecurityManager::~MamdaMultiSecurityManager () [virtual]

Destructor.

### 6.45.3 Member Function Documentation

#### 6.45.3.1 void Wombat::MamdaMultiSecurityManager::addHandler (MamdaMultiSecurityHandler ∗ handler)

Add a specialized handler for notifications about each security in the group. The handler is responsible for initializing any data structures and listeners for each security. Currently, only one handler can be registered.

#### 6.45.3.2 void Wombat::MamdaMultiSecurityManager::addSecurityListener (MamdaMsgListener ∗ listener, const char ∗ securitySymbol)

Add a specialized message listener (e.g., a MamdaQuoteListener, MamdaTradeListener, etc.) for a security.

#### 6.45.3.3 void Wombat::MamdaMultiSecurityManager::onMsg (MamdaSubscription ∗ subscription, const MamaMsg & msg, short msgType) [virtual]

Implementation of MamdaMsgListener interface. Implements Wombat::MamdaMsgListener.

The documentation for this class was generated from the following file:
• MamdaMultiSecurityManager.h
6.46 Wombat::MamdaNewsFields Class Reference

#include <MamdaNewsFields.h>

Static Public Member Functions

- static void setDictionary (const MamaDictionary &dictionary)
- static void reset ()
- static uint16_t getMaxFid ()
- static bool isSet ()

Static Public Attributes

- static const MamaFieldDescriptor * HEADLINE_TEXT
- static const MamaFieldDescriptor * HEADLINE_ID
- static const MamaFieldDescriptor * STORY_TEXT
- static const MamaFieldDescriptor * STORY_ID
- static const MamaFieldDescriptor * ORIG_STORY_ID
- static const MamaFieldDescriptor * SOURCE_ID
- static const MamaFieldDescriptor * ORIG_SOURCE_ID
- static const MamaFieldDescriptor * LANGUAGE_ID
- static const MamaFieldDescriptor * NATIVE_CODES
- static const MamaFieldDescriptor * NATIVE_SYMBOLS
- static const MamaFieldDescriptor * INDUSTRIES
- static const MamaFieldDescriptor * MARKET_SECTORS
- static const MamaFieldDescriptor * REGIONS
- static const MamaFieldDescriptor * COUNTRIES
- static const MamaFieldDescriptor * PRODUCTS
- static const MamaFieldDescriptor * TOPICS
- static const MamaFieldDescriptor * MISC_CODES
- static const MamaFieldDescriptor * SYMBOLS
- static const MamaFieldDescriptor * PRIORITY
- static const MamaFieldDescriptor * TIME
- static const MamaFieldDescriptor * REVISION_NUM
- static const MamaFieldDescriptor * STORY_STATUS
- static const MamaFieldDescriptor * STORY_TIME
- static const MamaFieldDescriptor * ORIG STORY_TIME
- static const MamaFieldDescriptor * STORY_HEADLINES
- static const MamaFieldDescriptor * HAS STORY
6.46.1 Member Function Documentation

6.46.1.1 static void Wombat::MamdaNewsFields::setDictionary (const MamaDictionary & dictionary) [static]

6.46.1.2 static void Wombat::MamdaNewsFields::reset () [static]

6.46.1.3 static uint16_t Wombat::MamdaNewsFields::getMaxFid () [static]

6.46.1.4 static bool Wombat::MamdaNewsFields::isSet () [static]

6.46.2 Member Data Documentation

6.46.2.1 const MamaFieldDescriptor* Wombat::MamdaNewsFields::HEADLINE_TEXT [static]

6.46.2.2 const MamaFieldDescriptor* Wombat::MamdaNewsFields::HEADLINE_ID [static]

6.46.2.3 const MamaFieldDescriptor* Wombat::MamdaNewsFields::STORY_TEXT [static]

6.46.2.4 const MamaFieldDescriptor* Wombat::MamdaNewsFields::STORY_ID [static]

6.46.2.5 const MamaFieldDescriptor* Wombat::MamdaNewsFields::ORIG_STORY_ID [static]

6.46.2.6 const MamaFieldDescriptor* Wombat::MamdaNewsFields::SOURCE_ID [static]

6.46.2.7 const MamaFieldDescriptor* Wombat::MamdaNewsFields::ORIG_SOURCE_ID [static]

6.46.2.8 const MamaFieldDescriptor* Wombat::MamdaNewsFields::LANGUAGE_ID [static]

6.46.2.9 const MamaFieldDescriptor* Wombat::MamdaNewsFields::NATIVE_CODES [static]

6.46.2.10 const MamaFieldDescriptor* Wombat::MamdaNewsFields::NATIVE_SYMBOLS [static]

6.46.2.11 const MamaFieldDescriptor* Wombat::MamdaNewsFields::INDUSTRIES [static]

6.46.2.12 const MamaFieldDescriptor* Wombat::MamdaNewsFields::MARKET_SECTORS [static]

6.46.2.13 const MamaFieldDescriptor* Wombat::MamdaNewsFields::REGIONS [static]

6.46.2.14 const MamaFieldDescriptor* Wombat::MamdaNewsFields::COUNTRIES [static]
• MamdaNewsFields.h
6.47 Wombat::MamdaNewsHeadline Class Reference

MamdaNewsHeadline represents a news headline and includes information about many types of meta-data attributes associated with the headline.

```cpp
#include <MamdaNewsHeadline.h>
```

Inheritance diagram for Wombat::MamdaNewsHeadline:

```
Wombat::MamdaBasicEvent
   |
   └── Wombat::MamdaNewsMetaData
        └── Wombat::MamdaNewsHeadline
```

Public Member Functions

- **MamdaNewsHeadline ()**
  * Default constructor.

- **MamdaNewsHeadline (const MamdaNewsHeadline &copy)**
  * Copy constructor.

- **virtual ~MamdaNewsHeadline ()**
  * Destructor.

- **virtual MamdaNewsHeadline & operator= (const MamdaNewsHeadline &rhs)**
  * Assignment operator.

- **virtual const char * getHeadlineText () const**
  * Get headline text.

- **virtual MamdaNewsHeadlineId getHeadlineId () const**
  * Get headline ID.

- **virtual MamdaNewsStoryId getNewsStoryId () const**
  * Get news story ID.

- **virtual MamdaNewsStoryId getNewsOrigStoryId () const**
  * Get original news story ID.

- **virtual bool getNewsHasStory () const**
  * Check if news has story.

- **virtual const char * getNewsSourceId () const**
  * Get news source ID.

- **virtual const char * getNewsOrigSourceId () const**
  * Get original news source ID.

- **virtual const char * getLanguageId () const**
  * Get news language ID.

- **virtual void getNativeCodes (const char **&codes, mama_size_t &numCodes) const**
  * Get an array of native meta-data codes associated with this news story.

```
Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
```
• virtual void `getNativeRelatedSymbols` (const char **&symbols, mama_size_t &numSymbols) const
  
  *Get an array of native feed symbol codes associated with this news story.*

• virtual void `getIndustries` (const char **&industries, mama_size_t &numIndustries) const
  
  *Get an array of normalized industry codes associated with this news story.*

• virtual void `getMarketSectors` (const char **&marketSectors, mama_size_t &numMarketSectors) const
  
  *Get an array of normalized market sector codes associated with this news story.*

• virtual void `getRegions` (const char **&regions, mama_size_t &numRegions) const
  
  *Get an array of normalized region codes associated with this news story.*

• virtual void `getCountries` (const char **&countries, mama_size_t &numCountries) const
  
  *Get an array of ISO country codes associated with this news story.*

• virtual void `getProducts` (const char **&products, mama_size_t &numProducts) const
  
  *Get an array of normalized product codes associated with this news story.*

• virtual void `getTopics` (const char **&topics, mama_size_t &numTopics) const
  
  *Get an array of normalized topic (or "subject") codes associated with this news story.*

• virtual void `getMiscCodes` (const char **&miscCodes, mama_size_t &numMiscCodes) const
  
  *Get an array of normalized miscellaneous codes associated with this news story.*

• virtual void `getRelatedSymbols` (const char **&symbols, mama_size_t &numSymbols) const
  
  *Get an array of normalized symbol codes associated with this news story.*

• virtual `MamdaNewsPriority getNewsPriority()` const

• virtual `mama_u16_t getNewsStoryRevNumber()` const

• virtual const `MamaDateTime & getNewsOrigStoryTime()` const

• virtual const char * `getSymbol()` const
  
  *Get the instruments string symbol.*

• virtual const char * `getPartId()` const
Get the participant identifier.

- virtual const MamaDateTime & getSrcTime () const
  Get the source time.

- virtual const MamaDateTime & getActivityTime () const
  Get the activity time.

- virtual const MamaDateTime & getLineTime () const
  Get the line time.

- virtual const MamaDateTime & getSendTime () const
  Get the send time.

- virtual const MamaMsgQual & getMsgQual () const
  Get the message qualifier.

- virtual const MamaDateTime & getEventTime () const
  Get the event time.

- virtual const mama_seqnum_t getEventSeqNum () const
  Get the event sequence number.

- MamdaFieldState getSymbolFieldState () const
  Get the string symbol field state.

- MamdaFieldState getPartIdFieldState () const
  Get the participant identifier field state.

- MamdaFieldState getSrcTimeFieldState () const
  Get the source time field state.

- MamdaFieldState getActivityTimeFieldState () const
  Get the activity time field state.

- MamdaFieldState getLineTimeFieldState () const
  Get the line time of the update.

- MamdaFieldState getSendTimeFieldState () const
  Get the send time field state.

- MamdaFieldState getMsgQualFieldState () const
Get the message qualifier field state.

- `MamdaFieldState getEventTimeFieldState () const`
  
  Get the event time field state.

- `MamdaFieldState getEventSeqNumFieldState () const`
  
  Get the event sequence number field state.

- `void setHeadlineText (const char *headlineText)`
- `void setHeadlineId (const char *headlineId)`
- `void setStoryId (const char *storyId)`
- `void setHasStory (bool hasStory)`
- `void setSourceId (const char *source)`
- `void setOrigStoryId (const char *origStoryId)`
- `void setOrigSourceId (const char *origSourceId)`
- `void setLanguageId (const char *langId)`
- `void setNativeCodes (const char **codes, mama_size_t numCodes)`
- `void setNativeRelatedSymbols (const char **symbols, mama_size_t numSymbols)`
- `void setIndustries (const char **industries, mama_size_t numIndustries)`
- `void setMarketSectors (const char **marketSectors, mama_size_t numMarketSectors)`
- `void setRegions (const char **regions, mama_size_t numRegions)`
- `void setCountries (const char **countries, mama_size_t numCountries)`
- `void setProducts (const char **products, mama_size_t numProducts)`
- `void setTopics (const char **topics, mama_size_t numTopics)`
- `void setMiscCodes (const char **codes, mama_size_t numCodes)`
- `void setRelatedSymbols (const char **symbols, mama_size_t numSymbols)`
- `void setPriority (MamdaNewsPriority priority)`
- `void setStoryRevNumber (mama_u16_t storyRevNum)`
- `void setEventSeqNum (mama_seqnum_t seqNum)`
- `void setEventTime (const MamaDateTime &eventTime)`
- `void setOrigStoryTime (const MamaDateTime &storyTime)`
- `void setSrcTime (const MamaDateTime &srcTime)`
- `void setActivityTime (const MamaDateTime &activityTime)`
- `void setLineTime (const MamaDateTime &lineTime)`
- `void setSendTime (const MamaDateTime &sendTime)`
- `void setMsgQual (const MamaMsgQual &msgQualifier)`
- `void clear ()`
- `void setSubscriptionInfo (MamdaSubscription *subsc)`

Implementation: save the subscription info for the headline (so we know where to request stories from).
• void setSubscInfo (MamaQueue *queue, MamaSource *source)
• MamaQueue * getQueue () const
  
   Implementation: get the subscription info for the headline.

• MamaSource * getSource () const

6.47.1 Detailed Description

MamdaNewsHeadline represents a news headline and includes information about many
types of meta-data attributes associated with the headline.

Note on story IDs: all headlines have a story ID, even though no story may exist for the
headline. The story ID is generated in order to be able to index together all revisions
of a headline/story over time.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 Wombat::MamdaNewsHeadline::MamdaNewsHeadline ()

Default constructor.

6.47.2.2 Wombat::MamdaNewsHeadline::MamdaNewsHeadline (const
MamdaNewsHeadline & copy)

Copy constructor.

6.47.2.3 virtual Wombat::MamdaNewsHeadline::MamdaNewsHeadline ()
  [virtual]

Destructor.

6.47.3 Member Function Documentation

6.47.3.1 virtual MamdaNewsHeadline& Wombat::MamdaNews-
Headline::operator= (const MamdaNewsHeadline & rhs)
  [virtual]

Assignment operator.
6.47 Wombat::MamdaNewsHeadline Class Reference

6.47.3.2 virtual const char∗ Wombat::MamdaNewsHeadline::getHeadlineText () const [virtual]

Returns:

The text of the headline for the story.

Implements Wombat::MamdaNewsMetaData.

6.47.3.3 virtual MamdaNewsHeadlineId Wombat::MamdaNewsHeadline::getHeadlineId () const [virtual]

Returns:

The headline ID for this headline.

6.47.3.4 virtual MamdaNewsStoryId Wombat::MamdaNewsHeadline::getNewsStoryId () const [virtual]

Returns:

The NYSE Technologies ID of the news story.

6.47.3.5 virtual MamdaNewsStoryId Wombat::MamdaNewsHeadline::getNewsOrigStoryId () const [virtual]

Returns:

The data source story ID for this news story.

6.47.3.6 virtual bool Wombat::MamdaNewsHeadline::getNewsHasStory () const [virtual]

Returns:

Whether the headline has an associated story at this time.

6.47.3.7 virtual const char∗ Wombat::MamdaNewsHeadline::getNewsSourceId () const [virtual]

Returns:

The data source of the news story.
6.47.3.8 virtual const char* Wombat::MamdaNewsHeadline::getOrigSourceId () const [virtual] Returns:

The original data source of the news story (e.g., if the story was provided by a news aggregator).

Implements Wombat::MamdaNewsMetaData.

6.47.3.9 virtual const char* Wombat::MamdaNewsHeadline::getLanguageId () const [virtual] Returns:

The ANSI language ID of the news story.

Implements Wombat::MamdaNewsMetaData.

6.47.3.10 virtual void Wombat::MamdaNewsHeadline::getNativeCodes (const char **& codes, mama_size_t & numCodes) const [virtual] Get an array of native meta-data codes associated with this news story.

Implements Wombat::MamdaNewsMetaData.

6.47.3.11 virtual void Wombat::MamdaNewsHeadline::getNativeRelatedSymbols (const char **& symbols, mama_size_t & numSymbols) const [virtual] Get an array of native feed symbol codes associated with this news story.

Implements Wombat::MamdaNewsMetaData.

6.47.3.12 virtual void Wombat::MamdaNewsHeadline::getIndustries (const char **& industries, mama_size_t & numIndustries) const [virtual] Get an array of normalized industry codes associated with this news story.

Implements Wombat::MamdaNewsMetaData.
Get an array of normalized market sector codes associated with this news story.
Implements Wombat::MamdaNewsMetaData.

Get an array of normalized region codes associated with this news story.
Implements Wombat::MamdaNewsMetaData.

Get an array of ISO country codes associated with this news story.
Implements Wombat::MamdaNewsMetaData.

Get an array of normalized product codes associated with this news story.
Implements Wombat::MamdaNewsMetaData.

Get an array of normalized topic (or "subject") codes associated with this news story.
Implements Wombat::MamdaNewsMetaData.

Get an array of normalized miscellaneous codes associated with this news story.
Miscellaneous codes are those not categorized as industry, market sector, region, country or product codes.
6.47.3.19 virtual void Wombat::MamdaNewsHeadline::getRelatedSymbols
(const char **& symbols, mama_size_t & numSymbols) const
[virtual]

Get an array of normalized symbol codes associated with this news story.
Implements Wombat::MamdaNewsMetaData.

6.47.3.20 virtual MamdaNewsPriority Wombat::MamdaNewsHeadline::get-
NewsPriority () const [virtual]

Returns:
Whether the feed provider has designated this story as with normal priority or
"hot" (important) priority.

Implements Wombat::MamdaNewsMetaData.

6.47.3.21 virtual mama_u16_t Wombat::MamdaNews-
Headline::getNewsStoryRevNumber () const
[virtual]

Returns:
The revision number. Note: the value returned is zero if the data source does not
provide revision numbers.

Implements Wombat::MamdaNewsMetaData.

6.47.3.22 virtual const MamaDateTime& Wombat::Mamda-
NewsHeadline::getNewsOrigStoryTime () const
[virtual]

Returns:
The original publish time of the news story.

Implements Wombat::MamdaNewsMetaData.

6.47.3.23 virtual const char* Wombat::MamdaNewsHeadline::getSymbol ()
const [virtual]

Get the instruments string symbol.
Returns:
Symbol. This is the "well-known" symbol for the security, including any symbol-
ology mapping performed by the publisher.

Implements Wombat::MamdaBasicEvent.

6.47.3.24 virtual const char∗ Wombat::MamdaNewsHeadline::getPartId () const [virtual]

Get the participant identifier.

Returns:
Participant ID. This may be an exchange identifier, a market maker ID, etc., or
NULL (if this is not related to any specific participant).

Implements Wombat::MamdaBasicEvent.

6.47.3.25 virtual const MamaDateTime& Wombat::MamdaNewsHeadline::getSrcTime () const [virtual]

Get the source time.

Returns:
Source time. Typically, the exchange generated feed time stamp. This is often
the same as the "event time", because many feeds do not distinguish between the
actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.47.3.26 virtual const MamaDateTime& Wombat::MamdaNewsHeadline::getActivityTime () const [virtual]

Get the activity time.

Returns:
Activity time. A feed handler generated time stamp representing when the data
item was last updated.

Implements Wombat::MamdaBasicEvent.

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
Get the line time.

Returns:

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

Get the send time.

Returns:

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime().

Implements Wombat::MamdaBasicEvent.

Get the message qualifier.

Returns:

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.
6.47 Wombat::MamdaNewsHeadline Class Reference

6.47.3.30 virtual const MamaDateTime& Wombat::MamdaNewsHeadline::getEventTime () const [virtual]

Get the event time.

Returns:

Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.47.3.31 virtual mama_seqnum_t Wombat::MamdaNewsHeadline::getEventSeqNum () const [virtual]

Get the event sequence number.

Returns:

Source sequence number. The exchange generated sequence number.

Implements Wombat::MamdaBasicEvent.

6.47.3.32 MamdaFieldState Wombat::MamdaNewsHeadline::getSymbolFieldState () const [virtual]

Get the string symbol field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.47.3.33 MamdaFieldState Wombat::MamdaNewsHeadline::getPartIdFieldState () const [virtual]

Get the participant identifier field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.47.3.34  MamdaFieldState Wombat::MamdaNewsHeadline::getSrcTimeFieldState () const [virtual]

Get the source time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.47.3.35  MamdaFieldState Wombat::MamdaNewsHeadline::getActivityTimeFieldState () const [virtual]

Get the activity time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.47.3.36  MamdaFieldState Wombat::MamdaNewsHeadline::getLineTimeFieldState () const [virtual]

Get the line time of the update.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.47.3.37  MamdaFieldState Wombat::MamdaNewsHeadline::getSendTimeFieldState () const [virtual]

Get the send time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.47.3.38 MamdaFieldState Wombat::MamdaNewsHeadline::getMsgQualifierFieldState () const [virtual]

Get the message qualifier field state.

Returns:
Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.47.3.39 MamdaFieldState Wombat::MamdaNewsHeadline::getEventTimeFieldState () const [virtual]

Get the event time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.47.3.40 MamdaFieldState Wombat::MamdaNewsHeadline::getEventSequenceNumFieldState () const [virtual]

Get the event sequence number field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.47 Wombat::MamdaNewsHeadline Class Reference

6.47.3.41 void Wombat::MamdaNewsHeadline::setHeadlineText (const char * headlineText)

6.47.3.42 void Wombat::MamdaNewsHeadline::setHeadlineId (const char * headlineId)

6.47.3.43 void Wombat::MamdaNewsHeadline::setStoryId (const char * storyId)

6.47.3.44 void Wombat::MamdaNewsHeadline::setHasStory (bool hasStory)

6.47.3.45 void Wombat::MamdaNewsHeadline::setSourceId (const char * source)

6.47.3.46 void Wombat::MamdaNewsHeadline::setOrigStoryId (const char * storyId)

6.47.3.47 void Wombat::MamdaNewsHeadline::setOrigSourceId (const char * origSourceId)

6.47.3.48 void Wombat::MamdaNewsHeadline::setLanguageId (const char * langId)

6.47.3.49 void Wombat::MamdaNewsHeadline::setNativeCodes (const char ** codes, mama_size_t numCodes)

6.47.3.50 void Wombat::MamdaNewsHeadline::setNativeRelatedSymbols (const char ** symbols, mama_size_t numSymbols)

6.47.3.51 void Wombat::MamdaNewsHeadline::setIndustries (const char ** industries, mama_size_t numIndustries)

6.47.3.52 void Wombat::MamdaNewsHeadline::setMarketSectors (const char ** marketSectors, mama_size_t numMarketSectors)

6.47.3.53 void Wombat::MamdaNewsHeadline::setRegions (const char ** regions, mama_size_t numRegions)

6.47.3.54 void Wombat::MamdaNewsHeadline::setCountries (const char ** countries, mama_size_t numCountries)

6.47.3.55 void Wombat::MamdaNewsHeadline::setProducts (const char ** products, mama_size_t numProducts)

6.47.3.56 void Wombat::MamdaNewsHeadline::setTopics (const char ** topics, mama_size_t numTopics)

6.47.3.57 void Wombat::MamdaNewsHeadline::setMiscCodes (const char ** codes, mama_size_t numCodes)

6.47.3.58 void Wombat::MamdaNewsHeadline::setRelatedSymbols (const char ** symbols, mama_size_t numSymbols)

6.47.3.59 void Wombat::MamdaNewsHeadline::setPriority (MamdaNewsPriority priority)
6.47.3.71 void Wombat::MamdaNewsHeadline::setSubscInfo (MamaQueue * queue, MamaSource * source)

6.47.3.72 MamaQueue* Wombat::MamdaNewsHeadline::getQueue () const

Implementation: get the subscription info for the headline.

6.47.3.73 MamaSource* Wombat::MamdaNewsHeadline::getSource () const

The documentation for this class was generated from the following file:

- MamdaNewsHeadline.h
MamdaNewsHandler is an interface for applications that want to have an easy way to handle news headlines.

```
#include <MamdaNewsHeadlineHandler.h>
```

### Public Member Functions

- virtual void `onNewsHeadline(MamdaNewsManager &manager, const MamaMsg &msg, const MamdaNewsHeadline &headline, void *closure)=0`
  
  *Method invoked when a news headline is received.*

- virtual `~MamdaNewsHeadlineHandler()`

### Detailed Description

MamdaNewsHandler is an interface for applications that want to have an easy way to handle news headlines.

The interface defines callback methods for different types of trade-related events: trades, errors/cancels, corrections, recaps and closing reports.

### Constructor & Destructor Documentation

#### 6.48.2.1 virtual Wombat::MamdaNewsHeadlineHandler::~MamdaNewsHeadlineHandler () [virtual]

59 {};

### Member Function Documentation

#### 6.48.3.1 virtual void Wombat::MamdaNewsHeadlineHandler::onNewsHeadline (MamdaNewsManager &manager, const MamaMsg &msg, const MamdaNewsHeadline &headline, void *closure) [pure virtual]

Method invoked when a news headline is received.

**Parameters:**

- `manager` The news manager invoking the callback.
msg  The MamaMsg that triggered this invocation.
headline  Access to the headline details.
closure  Access to the user supplied closure data.

The documentation for this class was generated from the following file:

- MamdaNewsHeadlineHandler.h
6.49 Wombat::MamdaNewsManager Class Reference

MamdaNewsManager provides a class for managing access to streaming news headlines, headline queries, individual story queries, etc.

#include <MamdaNewsManager.h>

Public Member Functions

- MamdaNewsManager()
- virtual ~MamdaNewsManager()
- void addBroadcastHeadlineSource(MamaQueue *queue, MamaSource *source, const char *symbol, void *closure)
  Add a broadcast headline source to the news manager.
- void clearBroadcastHeadlineSources()
  Remove all broadcast headline sources from the news manager.
- void addBroadcastStorySource(MamaQueue *queue, MamaSource *source, const char *symbol, void *closure)
  Add a broadcast story source to the news manager.
- void clearBroadcastStorySources()
  Remove all broadcast story sources from the news manager.
- void addBroadcastHeadlineHandler(MamdaNewsHeadlineHandler *handler)
  Add a headline handler for broadcast headlines.
- void addBroadcastStoryHandler(MamdaNewsStoryHandler *handler)
  Add a story handler for broadcast stories.
- void addQualityHandler(MamdaQualityListener *handler)
  Add a quality data handler.
- void addErrorHandler(MamdaErrorListener *handler)
  Add an error handler.
- void clearQuerySources()
- void executeQuery(MamaQueue *queue, MamaSource *source, MamdaQuery *newsQuery, MamdaNewsQueryType queryType, MamdaNewsQueryHandler *handler, void *closure)
  Request a story (by headline), invoking the given story handler when the story arrives (or not).
• void removeQuery (MamdaQuery *newsQuery)

• void requestStory (const MamdaNewsHeadline &headline, MamdaNewsStoryHandler *handler, void *closure)
   
   Request a story (by headline), invoking the given story handler when the story arrives (or not).

• void requestStory (MamaQueue *queue, MamaSource *source, MamdaNewsStoryId storyId, MamdaNewsStoryHandler *handler, void *closure)

   Request a story (by explicit headline ID, source and queue), invoking the given story handler when the story arrives (or not).

• void requestStoryLater (const MamdaNewsHeadline &headline, MamdaNewsStoryHandler *handler, double secondsLater, void *closure)

   Request a story (by headline) after some delay, invoking the given story handler when the story arrives (or not).

• void requestStoryLater (const MamdaNewsStory &story, MamdaNewsStoryHandler *handler, double secondsLater, void *closure)

   Request a story (by existing or partial story) after some delay, invoking the given story handler when the story arrives (or not).

• void requestStoryLater (MamaQueue *queue, MamaSource *source, MamdaNewsStoryId storyId, MamdaNewsStoryHandler *handler, double secondsLater, void *closure)

   Request a story (by headline) after some delay, invoking the given story handler when the story arrives (or not).

Public Attributes

• MamdaNewsManagerImpl & mImpl

6.49.1 Detailed Description

MamdaNewsManager provides a class for managing access to streaming news headlines, headline queries, individual story queries, etc.

Like the MAMDA "listener" classes, such as MamdaQuoteListener and MamdaTradeListener, the MamdaNewsManager cache certain information about the news service. Unlike those MAMDA "listener" classes, the MamdaNewsManager is also responsible for creating and managing its own subscriptions.
6.49 Wombat::MamdaNewsManager Class Reference

6.49.2 Constructor & Destructor Documentation

6.49.2.1 Wombat::MamdaNewsManager::MamdaNewsManager()

6.49.2.2 virtual Wombat::MamdaNewsManager::~MamdaNewsManager()
    [virtual]

6.49.3 Member Function Documentation

6.49.3.1 void Wombat::MamdaNewsManager::addBroadcastHeadlineSource(MamaQueue * queue, MamaSource * source, const char * symbol, void * closure)

Add a broadcast headline source to the news manager.
Headline updates will be passed to the handler(s) registered by addBroadcastHeadlineHandler().

6.49.3.2 void Wombat::MamdaNewsManager::clearBroadcastHeadlineSources()

Remove all broadcast headline sources from the news manager.

6.49.3.3 void Wombat::MamdaNewsManager::addBroadcastStorySource(MamaQueue * queue, MamaSource * source, const char * symbol, void * closure)

Add a broadcast story source to the news manager.
Stories will be passed to the handler(s) registered by addBroadcastStoryHandler().

6.49.3.4 void Wombat::MamdaNewsManager::clearBroadcastStorySources()

Remove all broadcast story sources from the news manager.

6.49.3.5 void Wombat::MamdaNewsManager::addBroadcastHeadlineHandler(MamdaNewsHeadlineHandler * handler)

Add a headline handler for broadcast headlines.
Add a story handler for broadcast stories.

Add a quality data handler.

Add an error handler.

Request a story (by headline), invoking the given story handler when the story arrives (or not).

Request a story (by headline), invoking the given story handler when the story arrives (or not).

Request a story (by explicit headline ID, source and queue), invoking the given story handler when the story arrives (or not).
The source and queue are the same parameters that correspond to the headline source (see addHeadlineSource()).

### 6.49.3.14 void Wombat::MamdaNewsManager::requestStoryLater (const MamdaNewsHeadline & headline, MamdaNewsStoryHandler * handler, double secondsLater, void * closure)

Request a story (by headline) after some delay, invoking the given story handler when the story arrives (or not).

### 6.49.3.15 void Wombat::MamdaNewsManager::requestStoryLater (const MamdaNewsStory & story, MamdaNewsStoryHandler * handler, double secondsLater, void * closure)

Request a story (by existing or partial story) after some delay, invoking the given story handler when the story arrives (or not).

### 6.49.3.16 void Wombat::MamdaNewsManager::requestStoryLater (MamaQueue * queue, MamaSource * source, MamdaNewsStoryId storyId, MamdaNewsStoryHandler * handler, double secondsLater, void * closure)

Request a story (by headline) after some delay, invoking the given story handler when the story arrives (or not).

### 6.49.4 Member Data Documentation

#### 6.49.4.1 MamdaNewsManagerImpl& Wombat::MamdaNewsManager::mImpl

The documentation for this class was generated from the following file:

- MamdaNewsManager.h
6.50  Wombat::MamdaNewsMetaData Class Reference

MamdaNewsMetaData represents information about many types of attributes associated with the news headline.

#include <MamdaNewsMetaData.h>

Inheritance diagram for Wombat::MamdaNewsMetaData:

```
Wombat::MamdaBasicEvent
Wombat::MamdaNewsMetaData
Wombat::MamdaNewsHeadline
```

Public Member Functions

* virtual const char * getHeadlineText () const =0
* virtual const char * getNewsSourceId () const =0
* virtual const char * getNewsOrigSourceId () const =0
* virtual const char * getLanguageId () const =0
* virtual void getNativeCodes (const char **&codes, mama_size_t &numCodes) const =0
  
  Get an array of native meta-data codes associated with this news story.

* virtual void getNativeRelatedSymbols (const char **&symbols, mama_size_t &numSymbols) const =0
  
  Get an array of native feed symbol codes associated with this news story.

* virtual void getIndustries (const char **&industries, mama_size_t &numIndustries) const =0
  
  Get an array of normalized industry codes associated with this news story.

* virtual void getMarketSectors (const char **&marketSectors, mama_size_t &numMarketSectors) const =0
  
  Get an array of normalized market sector codes associated with this news story.

* virtual void getRegions (const char **&regions, mama_size_t &numRegions) const =0
  
  Get an array of normalized region codes associated with this news story.
• virtual void get Countries (const char **&countries, mama_size_t &numCountries) const =0
  
  Get an array of ISO country codes associated with this news story.

• virtual void getTopics (const char **&topics, mama_size_t &numTopics) const =0
  
  Get an array of normalized topic (or "subject") codes associated with this news story.

• virtual void getProducts (const char **&products, mama_size_t &numProducts) const =0
  
  Get an array of normalized product codes associated with this news story.

• virtual void getMiscCodes (const char **&miscCodes, mama_size_t &numMiscCodes) const =0
  
  Get an array of normalized miscellaneous codes associated with this news story.

• virtual void getRelatedSymbols (const char **&symbols, mama_size_t &numSymbols) const =0
  
  Get an array of normalized symbol codes associated with this news story.

• virtual MamdaNewsPriority getNewsPriority () const =0
• virtual mama_u16_t getNewsStoryRevNumber () const =0
• virtual const MamaDateTime & getNewsOrigStoryTime () const =0
• virtual ~MamdaNewsMetaData ()

6.50.1 Detailed Description

MamdaNewsMetaData represents information about many types of attributes associated with the news headline.

Note that some information is inherited from the MamdaBasicEvent, including the source time (the time the data source sent the headline) and the event time (the time the article was actually published). Depending upon the richness of information provided by a particular data source, the source time and event time may be the same.

6.50.2 Constructor & Destructor Documentation

6.50.2.1 virtual Wombat::MamdaNewsMetaData::~MamdaNewsMetaData ()
  [virtual]

159 ();
6.50.3 Member Function Documentation

6.50.3.1 virtual const char * Wombat::MamdaNewsMetaData::getHeadlineText () const [pure virtual]

Returns:
The text of the headline for the story.

Implemented in Wombat::MamdaNewsHeadline.

6.50.3.2 virtual const char * Wombat::MamdaNewsMetaData::getNewsSourceId () const [pure virtual]

Returns:
The data source ID of the news story.

Implemented in Wombat::MamdaNewsHeadline.

6.50.3.3 virtual const char * Wombat::MamdaNewsMetaData::getNewsOrigSourceId () const [pure virtual]

Returns:
The original data source ID of the news story (e.g., if the story was provided by a
news aggregator).

Implemented in Wombat::MamdaNewsHeadline.

6.50.3.4 virtual const char * Wombat::MamdaNewsMetaData::getLanguageId () const [pure virtual]

Returns:
The ANSI language ID of the news story.

Implemented in Wombat::MamdaNewsHeadline.

6.50.3.5 virtual void Wombat::MamdaNewsMetaData::getNativeCodes (const char **& codes, mama_size_t & numCodes) const [pure virtual]

Get an array of native meta-data codes associated with this news story.

Implemented in Wombat::MamdaNewsHeadline.
6.50.3.6 virtual void Wombat::MamdaNewsMetaData::getNativeRelatedSymbols(const char **& symbols, mama_size_t & numSymbols) const [pure virtual]

Get an array of native feed symbol codes associated with this news story.
Implemented in Wombat::MamdaNewsHeadline.

6.50.3.7 virtual void Wombat::MamdaNewsMetaData::getIndustries(const char **& industries, mama_size_t & numIndustries) const [pure virtual]

Get an array of normalized industry codes associated with this news story.
Implemented in Wombat::MamdaNewsHeadline.

6.50.3.8 virtual void Wombat::MamdaNewsMetaData::getMarketSectors(const char **& marketSectors, mama_size_t & numMarketSectors) const [pure virtual]

Get an array of normalized market sector codes associated with this news story.
Implemented in Wombat::MamdaNewsHeadline.

6.50.3.9 virtual void Wombat::MamdaNewsMetaData::getRegions(const char **& regions, mama_size_t & numRegions) const [pure virtual]

Get an array of normalized region codes associated with this news story.
Implemented in Wombat::MamdaNewsHeadline.

6.50.3.10 virtual void Wombat::MamdaNewsMetaData::getCountries(const char **& countries, mama_size_t & numCountries) const [pure virtual]

Get an array of ISO country codes associated with this news story.
Implemented in Wombat::MamdaNewsHeadline.

6.50.3.11 virtual void Wombat::MamdaNewsMetaData::getTopics(const char **& topics, mama_size_t & numTopics) const [pure virtual]

Get an array of normalized topic (or "subject") codes associated with this news story.
Implemented in Wombat::MamdaNewsHeadline.
6.50.3.12 virtual void Wombat::MamdaNewsMetaData::getProducts (const char **& products, mama_size_t & numProducts) const [pure virtual]

Get an array of normalized product codes associated with this news story.
Implemented in Wombat::MamdaNewsHeadline.

6.50.3.13 virtual void Wombat::MamdaNewsMetaData::getMiscCodes (const char **& miscCodes, mama_size_t & numMiscCodes) const [pure virtual]

Get an array of normalized miscellaneous codes associated with this news story.
Miscellaneous codes are those not categorized as industry, market sector, region, country or product codes.
Implemented in Wombat::MamdaNewsHeadline.

6.50.3.14 virtual void Wombat::MamdaNewsMetaData::getRelatedSymbols (const char **& symbols, mama_size_t & numSymbols) const [pure virtual]

Get an array of normalized symbol codes associated with this news story.
Implemented in Wombat::MamdaNewsHeadline.

6.50.3.15 virtual MamdaNewsPriority Wombat::MamdaNewsMetaData::getNewsPriority () const [pure virtual]

Returns:
Whether the feed provider has designated this story as with normal priority or "hot" (important) priority.
Implemented in Wombat::MamdaNewsHeadline.

6.50.3.16 virtual mama_u16_t Wombat::MamdaNewsMetaData::getNewsStoryRevNumber () const [pure virtual]

Returns:
The revision number. Note: the value returned is zero if the data source does not provide revision numbers.
6.50 Wombat::MamdaNewsMetaData Class Reference

Implemented in Wombat::MamdaNewsHeadline.

6.50.3.17 virtual const MamaDateTime& Wombat::MamdaNewsMetaData::getNewsOrigStoryTime () const [pure virtual]

Returns:

The original publish time of the news story.

Implemented in Wombat::MamdaNewsHeadline.

The documentation for this class was generated from the following file:

- MamdaNewsMetaData.h
6.51 Wombat::MamdaNewsQueryHandler Class Reference

MamdaNewsHandler is an interface for applications that want to have an easy way to handle news queries.

```cpp
#include <MamdaNewsQueryHandler.h>
```

Public Member Functions

- virtual void `onNewsQueryHeadline` (MamdaNewsManager &manager, const MamaMsg &msg, const MamdaNewsHeadline &headline, const MamdaQuery &query, void *closure)=0
  
  *Method invoked when a news headline is received.*

- virtual void `onNewsQueryError` (MamdaNewsManager &manager, MamaMsg *msg, const char *errorStr, const MamaStatus &status, const MamdaQuery &query, void *closure)=0

- virtual void `onNewsQueryComplete` (MamdaNewsManager &manager, const MamdaQuery &query, void *closure)

- virtual `~MamdaNewsQueryHandler` ()

6.51.1 Detailed Description

MamdaNewsHandler is an interface for applications that want to have an easy way to handle news queries.

The interface defines callback methods for different types of trade-related events: trades, errors/cancels, corrections, recaps and closing reports.

6.51.2 Constructor & Destructor Documentation

6.51.2.1 virtual Wombat::MamdaNewsQueryHandler::~MamdaNewsQueryHandler () [virtual]

```cpp
77 {};
```

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.51.3 Member Function Documentation

6.51.3.1 virtual void Wombat::MamdaNewsQueryHandler::onNewsQueryHeadline (MamdaNewsManager & manager, const MamaMsg & msg, const MamdaNewsHeadline & headline, const MamdaQuery & query, void * closure) [pure virtual]

Method invoked when a news headline is received.

Parameters:

- **manager** The news manager invoking the callback.
- **msg** The MamaMsg that triggered this invocation.
- **headline** Access to the headline details.
- **closure** Access to the user supplied closure data.

6.51.3.2 virtual void Wombat::MamdaNewsQueryHandler::onNewsQueryError (MamdaNewsManager & manager, MamaMsg * msg, const char * errorStr, const MamaStatus & status, const MamdaQuery & query, void * closure) [pure virtual]

6.51.3.3 virtual void Wombat::MamdaNewsQueryHandler::onNewsQueryComplete (MamdaNewsManager & manager, const MamdaQuery & query, void * closure) [virtual]

The documentation for this class was generated from the following file:

- MamdaNewsQueryHandler.h
6.52 Wombat::MamdaNewsStory Class Reference

MamdaNewsStory represents a complete text of a news story.

#include <MamdaNewsStory.h>

Public Types

- NO_STORY = 0
- FULL_STORY = 1
- FETCHING_STORY = 2
- DELAYED_STORY = 3
- NOT_FOUND = 4
- DELETED = 5
- EXPIRED = 6
- UNKNOWN = 99

enum Status {
  NO_STORY = 0, FULL_STORY = 1, FETCHING_STORY = 2, DELAYED_STORY = 3,
  NOT_FOUND = 4, DELETED = 5, EXPIRED = 6, UNKNOWN = 99
}

The Status enumeration provides information about the status of the news story and is provided in responses to news story requests.

Public Member Functions

- MamdaNewsStory ()
  Default constructor.

- MamdaNewsStory (const MamdaNewsStory &copy)
  Copy constructor.

- virtual ~MamdaNewsStory ()
  Destructor.

- virtual MamdaNewsStory & operator= (const MamdaNewsStory &rhs)
  Assignment operator.

- virtual void setStory (const char *storyText, const char *storyId, uint16_t storyRevNum, Status storyStatus, const MamaDateTime &storyTime, const MamaDateTime &storyOrigTime, const MamdaNewsHeadlineId *headlines, mama_size_t numHeadlines)
Set all of the news story attributes.

- virtual void setNewsStoryText (const char ∗storyText)
  
  Set the news story text.

- virtual const char ∗ getNewsStoryText () const

- virtual void setNewsStoryId (MamdaNewsStoryId storyId)
  
  Set the news story id.

- virtual MamdaNewsStoryId getNewsStoryId () const

- virtual void setNewsStoryRevNumber (uint16_t storyRevNum)
  
  Set the story revision number.

- virtual uint16_t getNewsStoryRevNumber () const

- virtual void setStatus (Status status)
  
  Set the story status.

- virtual Status getStatus () const

- virtual const MamaDateTime & getStoryTime () const

- virtual const MamaDateTime & getOrigStoryTime () const

- virtual void getHeadlines (const MamdaNewsHeadlineId ∗&headlines, mama_-size_t &numHeadlines) const

- void setSubscInfo (MamaQueue ∗queue, MamaSource ∗source)

- MamaQueue ∗ getQueue () const

- MamaSource ∗ getSource () const

**Static Public Member Functions**

- static Status toStatus (const char ∗strVal)
  
  Convert a string representation of a Status to the enumerated representation.

- static const char ∗ toString (Status status)
  
  Convert an enumerated representation of a Status to a string representation.

### 6.52.1 Detailed Description

MamdaNewsStory represents a complete text of a news story.
6.52.2  Member Enumeration Documentation

6.52.2.1  enum Wombat::MamdaNewsStory::Status

The Status enumeration provides information about the status of the news story and is provided in responses to news story requests.

NO_STORY indicates that there is currently no story for the headline. This may occur for feeds that provide "alert" headlines, either as the only headline or as a precursor to a full story.

FULL_STORY indicates that the complete story text is being provided in the current callback.

FETCHING_STORY indicates that the story is currently being fetched by the publisher. This is a temporary status. An additional callback will automatically be invoked when the full story is available.

DELAYED_STORY indicates that the story is not currently available but is expected at some time in the future. No additional callback will be automatically invoked.

NOT_FOUND indicates that the publisher does not currently have a story for this headline and cannot determine whether a story will arrive for the headline. No additional callback will be automatically invoked.

UNKNOWN indicates an unknown condition (should not happen).

Enumerator:

***NO_STORY

***FULL_STORY

***FETCHING_STORY

***DELAYED_STORY

***NOT_FOUND

***DELETED

***EXPIRED

***UNKNOWN

```cpp
69 {  
70 NO_STORY = 0,  
71 FULL_STORY = 1,  
72 FETCHING_STORY = 2,  
73 DELAYED_STORY = 3,  
74 NOT_FOUND = 4,  
75 DELETED = 5,  
76 EXPIRED = 6,  
77 UNKNOWN = 99  
78 };
```
6.52.3 Constructor & Destructor Documentation

6.52.3.1 Wombat::MamdaNewsStory::MamdaNewsStory()

Default constructor.

6.52.3.2 Wombat::MamdaNewsStory::MamdaNewsStory (const MamdaNewsStory &copy)

Copy constructor.

6.52.3.3 virtual Wombat::MamdaNewsStory::∼MamdaNewsStory ()

[virtual]

Destructor.

6.52.4 Member Function Documentation

6.52.4.1 virtual MamdaNewsStory & Wombat::MamdaNewsStory::operator= (const MamdaNewsStory &rhs) [virtual]

Assignment operator.

6.52.4.2 virtual void Wombat::MamdaNewsStory::setStory (const char *storyText, const char *storyId, uint16_t storyRevNum, Status storyStatus, const MamaDateTime &storyTime, const MamaDateTime &storyOrigTime, const MamdaNewsHeadlineId *headlines, mama_size_t numHeadlines) [virtual]

Set all of the news story attributes.

6.52.4.3 virtual void Wombat::MamdaNewsStory::setNewsStoryText (const char *storyText) [virtual]

Set the news story text.

6.52.4.4 virtual const char * Wombat::MamdaNewsStory::getNewsStoryText ()

const [virtual]

Returns:

The text of the story.
6.52.4.5 virtual void Wombat::MamdaNewsStory::setNewsStoryId
(MamdaNewsStoryId storyId) [virtual]

Set the news story id.

6.52.4.6 virtual MamdaNewsStoryId Wombat::MamdaNewsStory::getNewsStoryId () const [virtual]

Returns:
The unique (for the data source) story ID for this news story.

6.52.4.7 virtual void Wombat::MamdaNewsStory::setNewsStoryRevNumber
(uint16_t storyRevNum) [virtual]

Set the story revision number.

6.52.4.8 virtual uint16_t Wombat::MamdaNewsStory::getNewsStoryRevNumber () const [virtual]

Returns:
The revision number. Note: the value returned is zero if the data source does not provide revision numbers.

6.52.4.9 virtual void Wombat::MamdaNewsStory::setStatus (Status status) [virtual]

Set the story status.

6.52.4.10 virtual Status Wombat::MamdaNewsStory::getStatus () const [virtual]

Returns:
The story status.

6.52.4.11 virtual const MamaDateTime& Wombat::MamdaNewsStory::getStoryTime () const [virtual]

Returns:
The latest story update time (e.g. time of correction).
virtual const MamaDateTime& Wombat::MamdaNewsStory::getOrigStoryTime () const [virtual]

Returns:
The original story publish time.

virtual void Wombat::MamdaNewsStory::getHeadlines (const MamdaNewsHeadlineId ∗ headlines, mama_size_t & numHeadlines) const [virtual]

Returns:
All of the headline IDs associated with this news story.

static Status Wombat::MamdaNewsStory::toStatus (const char ∗ strVal) [static]

Convert a string representation of a Status to the enumerated representation.

static const char ∗ Wombat::MamdaNewsStory::toString (Status status) [static]

Convert an enumerated representation of a Status to a string representation.

void Wombat::MamdaNewsStory::setSubscInfo (MamaQueue ∗ queue, MamaSource ∗ source)

MamaQueue ∗ Wombat::MamdaNewsStory::getQueue () const

MamaSource ∗ Wombat::MamdaNewsStory::getSource () const

The documentation for this class was generated from the following file:

- MamdaNewsStory.h
MamdaNewsHandler is an interface for applications that want to have an easy way to handle news stories.

```cpp
#include <MamdaNewsStoryHandler.h>
```

### Public Member Functions

- **virtual void onNewsStory (MamdaNewsManager &manager, const MamaMsg &msg, const MamdaNewsStory &story, void *closure)=0**
  
  Method invoked when a news story response is received.

- **virtual ~MamdaNewsStoryHandler ()**

### Detailed Description

MamdaNewsHandler is an interface for applications that want to have an easy way to handle news stories.

The interface defines callback methods for different types of news story events.

Instances of `MamdaNewsStory` do not survive beyond the `onNewsStory()` callback and should be copied if they need to be preserved beyond that callback.

### Constructor & Destructor Documentation

- **virtual Wombat::MamdaNewsStoryHandler::~MamdaNewsStoryHandler () [virtual] 63 {}**

### Member Function Documentation

- **virtual void Wombat::MamdaNewsStoryHandler::onNewsStory (MamdaNewsManager &manager, const MamaMsg &msg, const MamdaNewsStory &story, void *closure) [pure virtual]**

  Method invoked when a news story response is received.

  **Parameters:**

  - `manager` The `MamdaNewsManager` that is managing this news service.
msg  The MamaMsg that triggered this invocation.
story  The requested news story.
closure  The closure argument to the story request.

The documentation for this class was generated from the following file:

•  MamdaNewsStoryHandler.h
6.54 Wombat::MamdaOptionChain Class Reference

MamdaOptionChain is a specialized class to represent market data option chains.
#include <MamdaOptionChain.h>

Public Member Functions

- MamdaOptionChain (const char *symbol)
  MamdaOptionChain Constructor.

- ~MamdaOptionChain ()
  MamdaOptionChain Destructor.

- void setSymbol (const char *symbol)
  Set the underlying symbol for the option chain.

- const char * getSymbol () const
  Get the underlying symbol for the option chain.

- void setUnderlyingQuoteListener (const MamdaQuoteListener *quoteListener)
  Set the underlying quote information.

- void setUnderlyingTradeListener (const MamdaTradeListener *tradeListener)
  Set the underlying trade information.

- const MamdaQuoteListener * getUnderlyingQuoteListener () const
  Get the underlying quote information.

- const MamdaTradeListener * getUnderlyingTradeListener () const
  Get the underlying trade information.

- void addContract (const char *contractSymbol, MamdaOptionContract *contract)
  Add an option contract.

- void processNewContractDetails (const char *contractSymbol, MamdaOptionContract *contract)
  Add the contract to value-added-structures such as Put and Call side mappings and expiration by strike set, etc.

- void removeContract (const char *contractSymbol)
Remove an option contract.

- double *getAtTheMoney* (MamdaOptionAtTheMoneyCompareType compareType)
  
  Determine the underlying price ("at the money"), based on the mode of calculation.

- void *getStrikesWithinPercent* (StrikeSet &strikeSet, double percentage, MamdaOptionAtTheMoneyCompareType compareType)
  
  Determine the set of strike prices that are included in a given percentage range of the underlying price.

- void *getStrikesWithinRangeSize* (StrikeSet &strikeSet, int rangeLength, MamdaOptionAtTheMoneyCompareType compareType)
  
  Determine the set of strike prices that are included in a given fixed size range of strikes surrounding the underlying price.

- bool *getIsPriceWithinPercentOfMoney* (double price, double percentage, MamdaOptionAtTheMoneyCompareType compareType)
  
  Determine whether some price (e.g.

- iterator *callIterator* ()
  
  Obtain an iterator to the beginning of the list of call options.

- iterator *putIterator* ()
  
  Obtain an iterator to the beginning of the list of put options.

- const_iterator *callIterator* () const
  
  Obtain an iterator to the beginning of the list of call options.

- const_iterator *putIterator* () const
  
  Obtain an iterator to the beginning of the list of put options.

- const MamdaOptionExpirationDateSet & *getAllExpirations* () const
  
  Return the set of all expiration dates (which can be iterated over one date at a time).

- void *dump* ()
  
  Dump the option chain to standard output.

**Classes**

- class const_iterator
- class iterator
6.54.1 Detailed Description

MamdaOptionChain is a specialized class to represent market data option chains. The class has capabilities to store the current state of an entire option chain, or a subset of the chain.

6.54.2 Constructor & Destructor Documentation

6.54.2.1 Wombat::MamdaOptionChain::MamdaOptionChain (const char * symbol)

MamdaOptionChain Constructor.

6.54.2.2 Wombat::MamdaOptionChain::~MamdaOptionChain ()

MamdaOptionChain Destructor.

6.54.3 Member Function Documentation

6.54.3.1 void Wombat::MamdaOptionChain::setSymbol (const char * symbol)

Set the underlying symbol for the option chain.

6.54.3.2 const char* Wombat::MamdaOptionChain::getSymbol () const

Get the underlying symbol for the option chain.

6.54.3.3 void Wombat::MamdaOptionChain::setUnderlyingQuoteListener (const MamdaQuoteListener * quoteListener)

Set the underlying quote information.

The MamdaQuoteListener object would likely be an object that gets automatically updated by events handled elsewhere.

6.54.3.4 void Wombat::MamdaOptionChain::setUnderlyingTradeListener (const MamdaTradeListener * tradeListener)

Set the underlying trade information.

The MamdaTradeListener object would likely be an object that gets automatically updated by events handled elsewhere.
6.54.3.5 const MamdaQuoteListener* Wombat::MamdaOptionChain::getUnderlyingQuoteListener ()
const

Get the underlying quote information.
Returns the object provided by setUnderlyingQuoteListener(), if any.

6.54.3.6 const MamdaTradeListener* Wombat::MamdaOptionChain::getUnderlyingTradeListener ()
const

Get the underlying trade information.
Returns the object provided by setUnderlyingTradeListener(), if any.

6.54.3.7 void Wombat::MamdaOptionChain::addContract (const char* contractSymbol, MamdaOptionContract* contract)

Add an option contract.
This method would not normally be invoked by a user application. Rather, MamdaOptionChainListener would be most likely to call this method.

6.54.3.8 void Wombat::MamdaOptionChain::processNewContractDetails (const char* contractSymbol, MamdaOptionContract* contract)

Add the contract to value-added-structures such as Put and Call side mappings and expiration by strike set, etc.
Called by AddContract(const char*, MamdaOptionContract*). Can be called directly on a contract when additional mandatory fields are acquired for the contract that may enable it to be added to value-added-structures it wasn’t added to (due to the lack of information) when first added to the chain.

6.54.3.9 void Wombat::MamdaOptionChain::removeContract (const char* contractSymbol)

Remove an option contract.
This method would not normally be invoked by a user application. Rather, MamdaOptionChainListener would be most likely to call this method.
6.54.3.10  double Wombat::MamdaOptionChain::getAtTheMoney
            (MamdaOptionAtTheMoneyCompareType compareType)

Determine the underlying price ("at the money"), based on the mode of calculation.

6.54.3.11  void Wombat::MamdaOptionChain::getStrikesWithin-
            Percent (StrikeSet & strikeSet, double percentage,
            MamdaOptionAtTheMoneyCompareType compareType)

Determine the set of strike prices that are included in a given percentage range of the
underlying price.
If there are no strikes within the percentage range, then the set will be empty.

6.54.3.12  void Wombat::MamdaOptionChain::getStrikesWithin-
            RangeSize (StrikeSet & strikeSet, int rangeLength,
            MamdaOptionAtTheMoneyCompareType compareType)

Determine the set of strike prices that are included in a given fixed size range of strikes
surrounding the underlying price.
If rangeLen is odd, then the strike price nearest to the underlying price is treated as a
higher strike price. If rangeLen is even and the underlying price is exactly equal to a
strike price, then that strike price is treated as a higher strike price.

6.54.3.13  bool Wombat::MamdaOptionChain::getIsPriceWithin-
            PercentOfMoney (double price, double percentage,
            MamdaOptionAtTheMoneyCompareType compareType)

Determine whether some price (e.g. a strike price) is within a given percentage range of the underlying (at the money) price.

6.54.3.14  iterator Wombat::MamdaOptionChain::callIterator ()

Obtain an iterator to the beginning of the list of call options.
Use the iterator::hasNext() to test for a subsequent iterator and use iterator::next() method to move to it.

6.54.3.15  iterator Wombat::MamdaOptionChain::putIterator ()

Obtain an iterator to the beginning of the list of put options.
Use the `iterator::hasNext()` to test for a subsequent iterator and use `iterator::next()` method to move to it.

### 6.54.3.16 `const_iterator` Wombat::MamdaOptionChain::callIterator () const

Obtain an iterator to the beginning of the list of call options.

Use the `iterator::hasNext()` to test for a subsequent iterator and use `iterator::next()` method to move to it.

### 6.54.3.17 `const_iterator` Wombat::MamdaOptionChain::putIterator () const

Obtain an iterator to the beginning of the list of put options.

Use the `iterator::hasNext()` to test for a subsequent iterator and use `iterator::next()` method to move to it.

### 6.54.3.18 `const MamdaOptionExpirationDateSet&` Wombat::MamdaOptionChain::getAllExpirations () const

Return the set of all expiration dates (which can be iterated over one date at a time).

### 6.54.3.19 `void` Wombat::MamdaOptionChain::dump ()

Dump the option chain to standard output.

The documentation for this class was generated from the following file:

- `MamdaOptionChain.h`
6.55 Wombat::MamdaOptionChain::const_iterator
Class Reference

#include <MamdaOptionChain.h>

Public Member Functions

• \texttt{\texttilde{}const\_iterator} ()

• \texttt{const\_iterator} (const \texttt{const\_iterator} &)

• \texttt{const\_iterator \& operator=} (const \texttt{const\_iterator} &)

• bool \texttt{hasNext} ()

• const \texttt{MamdaOptionContract \* next} ()

Protected Member Functions

• \texttt{const\_iterator} (void *)

Protected Attributes

• \texttt{const\_Iterator\_impl \* mIter\_impl}

Friends

• class \texttt{MamdaOptionChain}
6.55 Wombat::MamdaOptionChain::const_iterator Class Reference

6.55.1 Constructor & Destructor Documentation

6.55.1.1 Wombat::MamdaOptionChain::const_iterator::~const_iterator ()

6.55.1.2 Wombat::MamdaOptionChain::const_iterator::const_iterator (const const_iterator &)

6.55.1.3 Wombat::MamdaOptionChain::const_iterator::const_iterator (void *)
[protected]

6.55.2 Member Function Documentation

6.55.2.1 const_iterator& Wombat::MamdaOptionChain::const_iterator::operator= (const const_iterator &)

6.55.2.2 bool Wombat::MamdaOptionChain::const_iterator::hasNext ()

6.55.2.3 const MamdaOptionContract* Wombat::MamdaOptionChain::const_iterator::next ()

6.55.3 Friends And Related Function Documentation

6.55.3.1 friend class MamdaOptionChain [friend]

6.55.4 Member Data Documentation

6.55.4.1 constIteratorImpl* Wombat::MamdaOptionChain::const_iterator::m_IterImpl [protected]

The documentation for this class was generated from the following file:

- MamdaOptionChain.h
# Wombat::MamdaOptionChain::iterator Class Reference

```cpp
#include <MamdaOptionChain.h>
```

## Public Member Functions

- `~iterator ()`
- `iterator (const iterator &)`
- `iterator & operator= (const iterator &)`
- `bool hasNext ()`
- `MamdaOptionContract * next ()`

## Protected Member Functions

- `iterator (void *)`

## Protected Attributes

- `iteratorImpl * mIterImpl`

## Friends

- `class MamdaOptionChain`
6.56 Wombat::MamdaOptionChain::iterator Class Reference

6.56.1 Constructor & Destructor Documentation

6.56.1.1 Wombat::MamdaOptionChain::iterator::~iterator ()

6.56.1.2 Wombat::MamdaOptionChain::iterator::iterator (const iterator &)

6.56.1.3 Wombat::MamdaOptionChain::iterator::iterator (void *)

6.56.2 Member Function Documentation

6.56.2.1 iterator& Wombat::MamdaOptionChain::iterator::operator= (const iterator &)

6.56.2.2 bool Wombat::MamdaOptionChain::iterator::hasNext ()

6.56.2.3 MamdaOptionContract* Wombat::MamdaOptionChain::iterator::next ()

6.56.3 Friends And Related Function Documentation

6.56.3.1 friend class MamdaOptionChain [friend]

6.56.4 Member Data Documentation

6.56.4.1 iteratorImpl* Wombat::MamdaOptionChain::iterator::mIterImpl [protected]

The documentation for this class was generated from the following file:

- MamdaOptionChain.h
Subclasses of this interface can be registered with the `MamdaOptionChainListener` in order to receive callbacks whenever the state of the underlying option chain changes on receipt of options updates.

```cpp
#include <MamdaOptionChainHandler.h>
```

Inheritance diagram for Wombat::MamdaOptionChainHandler::

```
Wombat::MamdaOptionChainHandler
   `-- Wombat::MamdaOptionChainView
```

### Public Member Functions

- **virtual void** `onOptionChainRecap(MamdaSubscription *subscription, MamdaOptionChainListener &listener, const MamaMsg &msg, MamdaOptionChain &chain)=0`  
  
  *Method invoked when an updated full option chain is available.*

- **virtual void** `onOptionContractCreate(MamdaSubscription *subscription, MamdaOptionChainListener &listener, const MamaMsg &msg, MamdaOptionContract &contract, MamdaOptionChain &chain)=0`  
  
  *Method invoked when a new contract is created in the option chain.*

- **virtual void** `onOptionSeriesUpdate(MamdaSubscription *subscription, MamdaOptionChainListener &listener, const MamaMsg &msg, const MamdaOptionSeriesUpdate &event, MamdaOptionChain &chain)=0`  
  
  *Method invoked upon when a new contract is added to or removed from the option chain, excluding upon receipt of the initial value.*

- **virtual void** `onOptionChainGap(MamdaSubscription *subscription, MamdaOptionChainListener &listener, const MamaMsg &msg, MamdaOptionChain &chain)=0`  
  
  *Method invoked when a gap in option chain updates is discovered.*

- **virtual** `~MamdaOptionChainHandler()`
6.57 Wombat::MamdaOptionChainHandler Class Reference

6.57.1 Detailed Description

Subclasses of this interface can be registered with the MamdaOptionChainListener in order to receive callbacks whenever the state of the underlying option chain changes on receipt of options updates.

6.57.2 Constructor & Destructor Documentation

6.57.2.1 virtual Wombat::MamdaOptionChainHandler::~MamdaOptionChainHandler () [virtual]

126 {};

6.57.3 Member Function Documentation

6.57.3.1 virtual void Wombat::MamdaOptionChainHandler::onOptionChainRecap (MamdaSubscription * subscription, MamdaOptionChainListener & listener, const MamaMsg & msg, MamdaOptionChain & chain) [pure virtual]

Method invoked when an updated full option chain is available.

The reason for the invocation may be any of the following:

- Initial image.
- Recap update (e.g., after server fault tolerant event or data quality event.)
- After stale status removed.

Parameters:

subscription The subscription which received the update.

listener The listener which invoked this callback.

msg The MamaMsg that triggered this invocation.

chain The complete option chain.

Implemented in Wombat::MamdaOptionChainView.

6.57.3.2 virtual void Wombat::MamdaOptionChainHandler::onOptionContractCreate (MamdaSubscription * subscription, MamdaOptionChainListener & listener, const MamaMsg & msg, MamdaOptionContract & contract, MamdaOptionChain & chain) [pure virtual]

Method invoked when a new contract is created in the option chain.
This method gets invoked exactly once for every option contract in the chain. The primary purpose of this method is to allow a user application to initialize any per-contract data as well to register handlers for trades and quotes.

Note: This method differs from onOptionSeriesUpdate() as follows: onOptionContractCreate() gets invoked every time a contract is added, even for the initial value; onOptionSeriesUpdate() is intended to report especially interesting events and is only invoked when a contract is added/removed after the initial value has been received.

Parameters:

- **subscription** The subscription which received the update.
- **listener** The listener which invoked this callback.
- **msg** The MamaMsg that triggered this invocation.
- **contract** The newly created option contract.
- **chain** The complete option chain.

Implemented in Wombat::MamdaOptionChainView.

### 6.57.3.3 virtual void Wombat::MamdaOptionChainHandler::onOptionSeriesUpdate(MamdaSubscription * subscription, MamdaOptionChainListener & listener, const MamaMsg & msg, const MamdaOptionSeriesUpdate & event, MamdaOptionChain & chain) [pure virtual]

Method invoked upon when a new contract is added to or removed from the option chain, excluding upon receipt of the initial value.

This method is typically invoked inly for special events, such as when options are added intraday or when options expire. Note: onOptionContractCreate() is also invoked when an option is added intraday.

Parameters:

- **subscription** The subscription which received the update.
- **listener** The listener which invoked this callback.
- **msg** The MamaMsg that triggered this invocation.
- **event** Access to the series update event details.
- **chain** The complete option chain.

See also:

- onOptionContractCreate.

Implemented in Wombat::MamdaOptionChainView.
Method invoked when a gap in option chain updates is discovered.

**Parameters:**

- `subscription` The `MamdaSubscription` handle.
- `listener` The listener which invoked this callback.
- `msg` The `MamaMsg` that triggered this invocation.
- `chain` The complete option chain.

Implemented in `Wombat::MamdaOptionChainView`.

The documentation for this class was generated from the following file:

- `MamdaOptionChainHandler.h`
6.58 Wombat::MamdaOptionChainListener Class Reference

MamdaOptionChainListener is a class that specializes in handling and managing option chain updates.

#include <MamdaOptionChainListener.h>

Inheritance diagram for Wombat::MamdaOptionChainListener:

```
Wombat::MamdaBasicEvent
<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
</table>
Wombat::MamdaMsgListener  Wombat::MamdaOptionSeriesUpdate

Wombat::MamdaOptionChainListener
```

Public Member Functions

- **MamdaOptionChainListener (const char *underlyingSymbol)**
  Create a specialized option chain listener.

- **MamdaOptionChainListener (MamdaOptionChain *chain)**
  Create a specialized option chain listener.

- **virtual ~MamdaOptionChainListener ()**
  Destructor.

- **void setMandatoryFields (bool expireDate, bool strikePrice, bool putCall)**
  Call this function to control whether or not option contracts, which are missing expire-date, strikePrice, or putCall fields are exposed to the user app by calling the registered onOptionContractCreate().

- **void addHandler (MamdaOptionChainHandler *handler)**
  Add a specialized option chain handler.

- **MamdaOptionChain & getOptionChain ()**
  Return the option chain associated with this listener.

- **const char * getSymbol () const**
  Get the instruments string symbol.
• const char * getPartId () const
    Get the participant identifier.

• void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)
    Implementation of MamdaListener interface.

• const MamaDateTime & getSrcTime () const
    Get the source time.

• const MamaDateTime & getActivityTime () const
    Get the activity time.

• const MamaDateTime & getLineTime () const
    Get the line time.

• const MamaDateTime & getSendTime () const
    Get the send time.

• const MamaMsgQual & getMsgQual () const
    Get the message qualifier.

• const MamaDateTime & getEventTime () const
    Get the event time.

• mama_seqnum_t getEventSeqNum () const
    Get the event sequence number.

• MamdaOptionContract * getOptionContract () const
• char getOptionAction () const
    Return the action related to the last series update message.

• MamdaFieldState getSymbolFieldState () const
    Get the string symbol field state.

• MamdaFieldState getPartIdFieldState () const
    Get the participant identifier field state.

• MamdaFieldState getEventSeqNumFieldState () const
    Get the event sequence number field state.

• MamdaFieldState getEventTimeFieldState () const
Get the event time field state.

- MamdaFieldState getSrcTimeFieldState () const
  Get the source time field state.

- MamdaFieldState getActivityTimeFieldState () const
  Get the activity time field state.

- MamdaFieldState getLineTimeFieldState () const
  Get the line time of the update.

- MamdaFieldState getSendTimeFieldState () const
  Get the send time field state.

- MamdaFieldState getMsgQualFieldState () const
  Get the message qualifier field state.

### 6.58.1 Detailed Description

**MamdaOptionChainListener** is a class that specializes in handling and managing option chain updates.

Developers provide their own implementation of the **MamdaOptionChainHandler** interface and will be delivered notifications for various types of options-related events.

### 6.58.2 Constructor & Destructor Documentation

#### 6.58.2.1 Wombat::MamdaOptionChainListener::MamdaOptionChainListener

(const char * underlyingSymbol)

Create a specialized option chain listener.

This listener handles option chain updates.

**Parameters:**

`underlyingSymbol` The underlying symbol for the option chain.

#### 6.58.2.2 Wombat::MamdaOptionChainListener::MamdaOptionChainListener

(MamdaOptionChain * chain)

Create a specialized option chain listener.
This listener handles option chain updates.

**Parameters:**

- **chain** The option chain object.

**6.58.2.3 virtual Wombat::MamdaOptionChainListener::~MamdaOptionChainListener () [virtual]**

Destructor.

### 6.58.3 Member Function Documentation

#### 6.58.3.1 void Wombat::MamdaOptionChainListener::setMandatoryFields (bool expireDate, bool strikePrice, bool putCall)

Call this function to control whether or not option contracts, which are missing expireDate, strikePrice, or putCall fields are exposed to the user app by calling the registered onOptionContractCreate().

.) callbacks. Passing a value of false will allow contracts without the particular field to be exposed, passing a value of true will ensure options without this field will not reach the user app. By default all three fields are considered mandatory.

**Parameters:**

- **expireDate** Whether the expiration date is mandatory.
- **strikePrice** Whether the strikePrice is mandatory.
- **putCall** Whether the put/call indicator is mandatory.

#### 6.58.3.2 void Wombat::MamdaOptionChainListener::addHandler (MamdaOptionChainHandler * handler)

Add a specialized option chain handler.

Currently, only one handler can (and must) be registered.

**Parameters:**

- **handler** An instance of the handler which will receive callback updates for activity on the option chain.
6.58.3.3 **MamdaOptionChain& Wombat::MamdaOptionChainListener::getOptionChain ()**

Return the option chain associated with this listener.

**Returns:**

The option chain being maintained by this listener instance.

6.58.3.4 **const char* Wombat::MamdaOptionChainListener::getSymbol () const [virtual]**

Get the instruments string symbol.

**Returns:**

Symbol. This is the "well-known" symbol for the security, including any symbology mapping performed by the publisher.

Implements **Wombat::MamdaBasicEvent**.

6.58.3.5 **const char* Wombat::MamdaOptionChainListener::getPartId () const [virtual]**

Get the participant identifier.

**Returns:**

Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).

Implements **Wombat::MamdaBasicEvent**.

6.58.3.6 **void Wombat::MamdaOptionChainListener::onMsg (MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [virtual]**

Implementation of MamdaListener interface.

Implements **Wombat::MamdaMsgListener**.
Get the source time.

**Returns:**

Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

Get the activity time.

**Returns:**

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements Wombat::MamdaBasicEvent.

Get the line time.

**Returns:**

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

Get the send time.
Returns:
Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime()).

Implements Wombat::MamdaBasicEvent.

6.58.3.11 const MamaMsgQual& Wombat::MamdaOptionChainListener::getMsgQual() const [virtual]

Get the message qualifier.

Returns:
Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.58.3.12 const MamaDateTime& Wombat::MamdaOptionChainListener::getEventTime() const [virtual]

Get the event time.

Returns:
Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.58.3.13 mama_seqnum_t Wombat::MamdaOptionChainListener::getEventSeqNum() const [virtual]

Get the event sequence number.

Returns:
Source sequence number. The exchange generated sequence number.

Implements Wombat::MamdaBasicEvent.
6.58.3.14 **MamdaOptionContract*** Wombat::MamdaOptionChainListener::getOptionContract () const

**Returns:**

The option contract to which the most recent event applies. If the contract is new, it will have already been added to the chain. If it is being removed, it will have already been removed from the chain.

Reimplemented from Wombat::MamdaOptionSeriesUpdate.

6.58.3.15 **char** Wombat::MamdaOptionChainListener::getOptionAction () const

Return the action related to the last series update message.

**Returns:**

The action related to the last series update message.

Reimplemented from Wombat::MamdaOptionSeriesUpdate.

6.58.3.16 **MamdaFieldState** Wombat::MamdaOptionChainListener::getSymbolFieldState () const [virtual]

Get the string symbol field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.58.3.17 **MamdaFieldState** Wombat::MamdaOptionChainListener::getParticipantIdFieldState () const [virtual]

Get the participant identifier field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.58.3.18 `MamdaFieldState` Wombat::MamdaOptionChainListener::getEventSeqNumFieldState () const
[virtual]

Get the event sequence number field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaBasicEvent**.

6.58.3.19 `MamdaFieldState` Wombat::MamdaOptionChainListener::getEventTimeFieldState () const
[virtual]

Get the event time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaBasicEvent**.

6.58.3.20 `MamdaFieldState` Wombat::MamdaOptionChainListener::getSrcTimeFieldState () const
[virtual]

Get the source time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaBasicEvent**.

6.58.3.21 `MamdaFieldState` Wombat::MamdaOptionChainListener::getActivityTimeFieldState () const
[virtual]

Get the activity time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaBasicEvent**.
MamdaFieldState Wombat::MamdaOptionChainListener::getLineTimeFieldState () const
[virtual]

Get the line time of the update.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

MamdaFieldState Wombat::MamdaOptionChainListener::getSendTimeFieldState () const
[virtual]

Get the send time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

MamdaFieldState Wombat::MamdaOptionChainListener::getMsgQualFieldState () const
[virtual]

Get the message qualifier field state.

Returns:

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

The documentation for this class was generated from the following file:

- MamdaOptionChainListener.h
A class that represents a "view" of a subset of an option chain.

```cpp
#include <MamdaOptionChainView.h>
```

Inheritance diagram for Wombat::MamdaOptionChainView:

- Wombat::MamdaOptionChainHandler
- Wombat::MamdaOptionChainView

### Public Member Functions

- **MamdaOptionChainView (MamdaOptionChain &chain)**
  
  Create a view on the given option chain.

- **virtual ~MamdaOptionChainView ()**

  Destructor.

- **const char * getSymbol () const**

  Return the symbol for the option chain.

- **void setAtTheMoneyType (MamdaOptionAtTheMoneyCompareType atTheMoneyType)**

  Set how the underlying price ("at the money") is determined.

- **void setStrikeRangePercent (double percentMargin)**

  Set the range of strike prices to be included in the view by percentage variation from the underlying price.

- **void setStrikeRangeNumber (int number)**

  Set the number of strike prices to be included in the view.

- **void setExpirationRangeDays (int expirationDays)**

  Set the range of expiration dates to be included in the view by the maximum number of days until expiration.

- **void setNumberOfExpirations (int numExpirations)**

  Set the number of expiration dates to be included in the view.
6.59 Wombat::MamdaOptionChainView Class Reference

- void setJitterMargin (double percentMargin)
  Set a "jitter margin" to avoid having the range jump between different strike prices when the underlying price hovers right on the edge of a range boundary.

- bool isVisible (const MamdaOptionContract &contract) const
  Return whether an option contract falls within this view's parameters.

- const MamdaOptionExpirationDateSet & getExpireDateSet () const
  Return the option expiration date set, which contains the subset of strike prices for the view, and the contracts within the strike price.

- void onOptionChainRecap (MamdaSubscription *subscription, MamdaOptionChainListener &listener, const MamaMsg &msg, MamdaOptionChain &chain)
  Handler for option chain recaps and initial values.

- void onOptionContractCreate (MamdaSubscription *subscription, MamdaOptionChainListener &listener, const MamaMsg &msg, MamdaOptionContract &contract, MamdaOptionChain &chain)
  Handler for option chain structural updates.

- void onOptionSeriesUpdate (MamdaSubscription *subscription, MamdaOptionChainListener &listener, const MamaMsg &msg, const MamdaOptionSeriesUpdate &event, MamdaOptionChain &chain)
  Handler for option chain structural updates.

- void onOptionChainGap (MamdaSubscription *subscription, MamdaOptionChainListener &listener, const MamaMsg &msg, MamdaOptionChain &chain)
  Handler for gaps.

6.59.1 Detailed Description

A class that represents a "view" of a subset of an option chain.

The view can be restricted to a percentage or number of strike prices around "the money" as well as to a maximum number of days into the future. The view will be adjusted to include strike prices within the range as the underlying price moves. This means that the range of strike prices will change over time. In order to avoid a "jitter" in the range of strike prices when the underlying price hovers right on the edge of a range boundary, the class also provides a "jitter margin" as some percentage of the underlying price (default is 0.5%).
6.59.2 Constructor & Destructor Documentation

6.59.2.1 Wombat::MamdaOptionChainView::MamdaOptionChainView
(MamdaOptionChain & chain)

Create a view on the given option chain.
Multiple views are supported on any given option chain.

Parameters:

chain The MamdaOptionChain on which to provide a view.

6.59.2.2 virtual Wombat::MamdaOptionChainView::~MamdaOptionChainView () [virtual]

Destructor.

6.59.3 Member Function Documentation

6.59.3.1 const char∗ Wombat::MamdaOptionChainView::getSymbol () const

Return the symbol for the option chain.

Returns:

The symbol for the option chain.

6.59.3.2 void Wombat::MamdaOptionChainView::setAtTheMoneyType
(MamdaOptionAtTheMoneyCompareType atTheMoneyType)

Set how the underlying price ("at the money") is determined.

Parameters:

atTheMoneyType The comparator enumeration to determine how at the money is
determined.

See also:

MamdaOptionAtTheMoneyCompareType
6.59.3.3  void Wombat::MamdaOptionChainView::setStrikeRangePercent (double percentMargin)

Set the range of strike prices to be included in the view by percentage variation from
the underlying price.

The range of strike prices in the view will vary as the underlying varies. The "jitter
margin" avoids switching between ranges too often.

Parameters:

  percentMargin  Percentage variation the strike price has from the underlying.

6.59.3.4  void Wombat::MamdaOptionChainView::setStrikeRangeNumber (int number)

Set the number of strike prices to be included in the view.

The "jitter margin" avoids switching between ranges too often.

Parameters:

  number  The number of strikes to include in the view.

6.59.3.5  void Wombat::MamdaOptionChainView::setExpirationRangeDays (int expirationDays)

Set the range of expiration dates to be included in the view by the maximum number
of days until expiration.

Note: a non-zero range overrides a specific number of expirations set by setNumber-
OfExpirations().

Parameters:

  expirationDays  The maximum number of days until expiration.

6.59.3.6  void Wombat::MamdaOptionChainView::setNumberOfExpirations (int numExpirations)

Set the number of expiration dates to be included in the view.

Note: a non-zero range (set by setExpirationRangeDays()) overrides a specific number
of expirations.
Parameters:

- **numExpirations** The number of expiration dates to include in the view.

### 6.59.3.7 void Wombat::MamdaOptionChainView::setJitterMargin (double percentMargin)

Set a "jitter margin" to avoid having the range jump between different strike prices when the underlying price hovers right on the edge of a range boundary. The underlying is allowed to fluctuate within the jitter margin without the range being reset. When the underlying moves beyond the jitter margin, the range is reset.

Parameters:

- **percentMargin** Percentage from the range boundary the underlying price can fluctuate without resetting the range.

### 6.59.3.8 bool Wombat::MamdaOptionChainView::isVisible (const MamdaOptionContract & contract) const

Return whether an option contract falls within this view’s parameters.

Parameters:

- **contract** The contract which is being checked for visibility by the current view.

### 6.59.3.9 const MamdaOptionExpirationDateSet & Wombat::MamdaOptionChainView::getExpireDateSet () const

Return the option expiration date set, which contains the subset of strike prices for the view, and the contracts within the strike price.

Returns:

The expiration date set for the option.

### 6.59.3.10 void Wombat::MamdaOptionChainView::onOptionChainRecap (MamdaSubscription * subscription, MamdaOptionChainListener & listener, const MamaMsg & msg, MamdaOptionChain & chain) [virtual]

Handler option chain recaps and initial values.
Parameters:

\begin{itemize}
\item \textit{subscription} The subscription which received the update.
\item \textit{listener} The listener instance which invoked this callback.
\item \textit{msg} The msg which resulted in the callback being invoked.
\item \textit{chain} The complete option chain.
\end{itemize}

Implements \texttt{Wombat::MamdaOptionChainHandler}.

\subsection*{6.59.3.11 \texttt{void Wombat::MamdaOptionChainView::onOptionContractCreate( MamdaSubscription \* subscription, MamdaOptionChainListener \& listener, const MamaMsg \& msg, MamdaOptionContract \& contract, MamdaOptionChain \& chain) \[virtual\]}

Handler for option chain structural updates.

Parameters:

\begin{itemize}
\item \textit{subscription} The subscription which received the update.
\item \textit{listener} The listener instance which invoked this callback.
\item \textit{msg} The msg which resulted in the callback being invoked.
\item \textit{contract} The newly created option contract.
\item \textit{chain} The complete option chain.
\end{itemize}

Implements \texttt{Wombat::MamdaOptionChainHandler}.

\subsection*{6.59.3.12 \texttt{void Wombat::MamdaOptionChainView::onOptionSeriesUpdate( MamdaSubscription \* subscription, MamdaOptionChainListener \& listener, const MamaMsg \& msg, const MamdaOptionSeriesUpdate \& event, MamdaOptionChain \& chain) \[virtual\]}

Handler for option chain structural updates.

Parameters:

\begin{itemize}
\item \textit{subscription} The subscription which received the update.
\item \textit{listener} The listener which invoked this callback.
\item \textit{msg} The msg which resulted in the callback being invoked.
\item \textit{event} Access to the series update event details.
\item \textit{chain} The complete option chain.
\end{itemize}

Implements \texttt{Wombat::MamdaOptionChainHandler}.
6.59.3.13  void Wombat::MamdaOptionChainView::onOptionChainGap
(MamdaSubscription * subscription, MamdaOptionChainListener
& listener, const MamaMsg & msg, MamdaOptionChain & chain)
  [virtual]

Handler for gaps.

Parameters:

  subscription  The subscription which received the update.
  listener      The listener which invoked this callback.
  msg           The msg which resulted in the callback being invoked.
  chain         The complete option chain.

Implements Wombat::MamdaOptionChainHandler.

The documentation for this class was generated from the following file:

- MamdaOptionChainView.h
6.60 Wombat::MamdaOptionChainViewRangeHandler Class Reference

Class to handle change in a MamdaOptionChainView range.
#include <MamdaOptionChainViewRangeHandler.h>

6.60.1 Detailed Description

Class to handle change in a MamdaOptionChainView range.
The documentation for this class was generated from the following file:

- MamdaOptionChainViewRangeHandler.h
6.61 Wombat::MamdaOptionContract Class Reference

A class that represents a single option contract.

#include <MamdaOptionContract.h>

Public Member Functions

- **MamdaOptionContract** (const char *symbol, const char *exchange, const MamaDateTime &expireDate, double strikePrice, MamdaOptionPutCall putCall)
  
  Constructor from expiration date, strike price, and put/call indicator.

- **MamdaOptionContract** (const char *symbol, const char *exchange)
  
  Constructor without expiration date, strike price, and put/call indicator.

- **~MamdaOptionContract** ()
  
  Destructor.

- void **setExpireDate** (const MamaDateTime &expireDate)
  
  Set the expiration date.

- void **setStrikePrice** (double strikePrice)
  
  Set the strike price.

- void **setPutCall** (MamdaOptionPutCall putCall)
  
  Set the put/call indicator.

- void **setOpenInterest** (uint32_t openInterest)
  
  Set the open interest size.

- void **setExerciseStyle** (MamdaOptionExerciseStyle exerciseStyle)
  
  Set the exercise style.

- void **setRecapRequired** (bool recapRequired)
  
  Set the recap required field.

- const char * **getSymbol** () const
  
  Return the OPRA contract symbol.

- const char * **getExchange** () const
Return the exchange.

- const MamaDateTime & getExpireDate () const
  Return the expiration date.

- const char * getExpireDateStr () const
  Return the expiration date as a string.

- bool gotExpireDate () const
  Return whether or not expiration date has been set.

- double getStrikePrice () const
  Return the strike price.

- bool gotStrikePrice () const
  Return whether or not strike price has been set.

- MamdaOptionPutCall getPutCall () const
  Return the put/call indicator.

- bool gotPutCall () const
  Return whether or not put call has been set.

- uint32_t getOpenInterest () const
  Return the level of interest at opening.

- bool gotOpenInterest () const
  Return whether or not open interest has been set.

- MamdaOptionExerciseStyle getExerciseStyle () const
  Return the style indicator - American(A)/European(E)/Capped(C).

- bool gotExerciseStyle () const
  Return whether or not exercise style has been set.

- bool getRecapRequired () const
  Get the recap required field.

- void addMsgListener (MamdaMsgListener *listener)
  Add a generic MamdaMsgListener to this option contract.

- void addTradeHandler (MamdaTradeHandler *handler)
Add a `MamdaTradeHandler` for handling trade updates to this option contract.

- `void addQuoteHandler (MamdaQuoteHandler *handler)`
  Add a `MamdaQuoteHandler` for handling quote updates to this option contract.

- `void addFundamentalHandler (MamdaFundamentalHandler *handler)`
  Add a `MamdaFundamentalHandler` for handling fundamental data fields.

- `void setCustomObject (void *object)`
  Add a custom object to this option contract.

- `MamdaTradeRecap & getTradeInfo () const`
  Return the current trade fields.

- `MamdaQuoteRecap & getQuoteInfo () const`
  Return the current quote fields.

- `MamdaFundamentals & getFundamentalsInfo () const`
  Return the current fundamentals fields.

- `void * getCustomObject () const`
  Return the custom object.

- `std::vector< MamdaMsgListener *> & getMsgListeners ()`
  Return the vector of message listeners.

- `MamdaTradeListener & getTradeListener ()`
  Return the trade listener.

- `MamdaQuoteListener & getQuoteListener ()`
  Return the quote listener.

- `MamdaFundamentalListener & getFundamentalListener ()`
  Return the fundamental listener.

- `void setInView (bool inView)`
  Set whether this contract is in the "view" within the option chain.

- `bool getInView ()`
  Return whether this contract is in the "view" within the option chain.

- `int64_t getSeqNum ()`
- `void setSeqNum (int64_t num)`
6.61 Wombat::MamdaOptionContract Class Reference

6.61.1 Detailed Description

A class that represents a single option contract.

Instances of this object are typically created by the MamdaOptionChainListener. Applications may attach a custom object to each instance of MamdaOptionContract.

Note: User applications can be notified of creation of MamdaOptionContract instances via the MamdaOptionChainListener.onOptionContractCreate() method.

Note: It is possible to provide individual MamdaTradeHandler and MamdaQuoteHandler handlers for trades and quotes, even though the MamdaOptionChainHandler also provides a general callback interface for updates to individual contracts.

6.61.2 Constructor & Destructor Documentation

6.61.2.1 Wombat::MamdaOptionContract::MamdaOptionContract (const char * symbol, const char * exchange, const MamaDateTime & expireDate, double strikePrice, MamdaOptionPutCall putCall)

Constructor from expiration date, strike price, and put/call indicator.

Parameters:
- symbol The option symbol.
- exchange The exchange identifier.
- expireDate The options expiration date.
- strikePrice The options strike price.
- putCall Whether this is a put or call option.

6.61.2.2 Wombat::MamdaOptionContract::MamdaOptionContract (const char * symbol, const char * exchange)

Constructor without expiration date, strike price, and put/call indicator.

Parameters:
- symbol The option symbol.
- exchange The exchange identifier.

6.61.2.3 Wombat::MamdaOptionContract::~MamdaOptionContract ()

Destructor.
6.61.3 Member Function Documentation

6.61.3.1 void Wombat::MamdaOptionContract::setExpireDate (const MamaDateTime & expireDate)

Set the expiration date.

Parameters:

expireDate  The options expiration date.

6.61.3.2 void Wombat::MamdaOptionContract::setStrikePrice (double strikePrice)

Set the strike price.

Parameters:

strikePrice  The options strike price.

6.61.3.3 void Wombat::MamdaOptionContract::setPutCall (MamdaOptionPutCall putCall)

Set the put/call indicator.

Parameters:

putCall  Whether this is a put or call option.

6.61.3.4 void Wombat::MamdaOptionContract::setOpenInterest (uint32_t openInterest)

Set the open interest size.

Parameters:

openInterest  The open interest for the option.
6.61.3.5 void Wombat::MamdaOptionContract::setExerciseStyle(MamdaOptionExerciseStyle exerciseStyle)

Set the exercise style.

**Parameters:**

*exerciseStyle* The exercise style for the option.

6.61.3.6 void Wombat::MamdaOptionContract::setRecapRequired (bool recapRequired)

Set the recap required field.
Typically only used by the MamdaOptionChainListener

**Parameters:**

*recapRequired* Whether a recap is required by the contract.

6.61.3.7 const char* Wombat::MamdaOptionContract::getSymbol () const

Return the OPRA contract symbol.

**Returns:**

The OPRA contract symbol.

6.61.3.8 const char* Wombat::MamdaOptionContract::getExchange () const

Return the exchange.

**Returns:**

The exchange identifier.

6.61.3.9 const MamaDateTime& Wombat::MamdaOptionContract::getExpirationDate () const

Return the expiration date.

**Returns:**

The options expiration date.
6.61.3.10  const char* Wombat::MamdaOptionContract::getExpireDateStr () const

Return the expiration date as a string.

Returns:

The options expiration date as a string.

6.61.3.11  bool Wombat::MamdaOptionContract::gotExpireDate () const

Return whether or not expiration date has been set.

Returns:

Whether the contract has an expiration date set.

6.61.3.12  double Wombat::MamdaOptionContract::getStrikePrice () const

Return the strike price.

Returns:

The option contracts strike price.

6.61.3.13  bool Wombat::MamdaOptionContract::gotStrikePrice () const

Return whether or not strike price has been set.

Returns:

Whether the strike price has been set for the contract.

6.61.3.14  MamdaOptionPutCall Wombat::MamdaOptionContract::getPutCall () const

Return the put/call indicator.

Returns:

Indicating whether this is a put or call option contract.
6.61.3.15 bool Wombat::MamdaOptionContract::gotPutCall () const

Return whether or not put call has been set.

Returns:

Whether the put/call indicator has been set for this contract.

6.61.3.16 uint32_t Wombat::MamdaOptionContract::getOpenInterest () const

Return the level of interest at opening.

Returns:

The opening interest for the option contract.

6.61.3.17 bool Wombat::MamdaOptionContract::gotOpenInterest () const

Return whether or not open interest has been set.

Returns:

Whether a value for opening interest has been set for the option contract.

6.61.3.18 MamdaOptionExerciseStyle Wombat::MamdaOptionContract::getExerciseStyle () const

Return the style indicator - American(A)/European(E)/Capped(C).

Returns:

The exercise style for the option contract.

6.61.3.19 bool Wombat::MamdaOptionContract::gotExerciseStyle () const

Return whether or not exercise style has been set.

Returns:

Whether the exercise style for the option contract has been set.
6.61.3.20  bool Wombat::MamdaOptionContract::getRecapRequired () const

Get the recap required field.
Reserved for use by the MamdaOptionChainListener.

Returns:
Whether a recap is required for this contract.

6.61.3.21  void Wombat::MamdaOptionContract::addMsgListener
(MamdaMsgListener * listener)

Add a generic MamdaMsgListener to this option contract.

Parameters:
listener  A instance of the listener interface.

6.61.3.22  void Wombat::MamdaOptionContract::addTradeHandler
(MamdaTradeHandler * handler)

Add a MamdaTradeHandler for handling trade updates to this option contract.

Parameters:
handler  A trade handler.

6.61.3.23  void Wombat::MamdaOptionContract::addQuoteHandler
(MamdaQuoteHandler * handler)

Add a MamdaQuoteHandler for handling quote updates to this option contract.

Parameters:
handler  A quote handler.

6.61.3.24  void Wombat::MamdaOptionContract::addFundamentalHandler
(MamdaFundamentalHandler * handler)

Add a MamdaFundamentalHandler for handling fundamental data fields.

Parameters:
handler  A fundamental handler.
6.61.3.25  void Wombat::MamdaOptionContract::setCustomObject (void * object)
Add a custom object to this option contract.
Such an object might contain customer per-contract data.

Parameters:

  object  Associate closure data with the option contract.

6.61.3.26  MamdaTradeRecap& Wombat::MamdaOptionContract::getTradeInfo () const
Return the current trade fields.

Returns:

  The current trade related fields for the option contract.

6.61.3.27  MamdaQuoteRecap& Wombat::MamdaOptionContract::getQuoteInfo () const
Return the current quote fields.

Returns:

  The current quote related fields for the option contract.

6.61.3.28  MamdaFundamentals& Wombat::MamdaOptionContract::getFundamentalsInfo () const
Return the current fundamentals fields.

Returns:

  The current fundamental fields for the option contract.

6.61.3.29  void* Wombat::MamdaOptionContract::getCustomObject () const
Return the custom object.

Returns:

  The closure.
6.61.3.30 std::vector<MamdaMsgListener*>& Wombat::MamdaOptionContract::getMsgListeners ()

Return the vector of message listeners.

Returns:
Vector of message listeners registered with the object.

6.61.3.31 MamdaTradeListener& Wombat::MamdaOptionContract::getTradeListener ()

Return the trade listener.

Returns:
The registered trade listener.

6.61.3.32 MamdaQuoteListener& Wombat::MamdaOptionContract::getQuoteListener ()

Return the quote listener.

Returns:
The registered quote listener.

6.61.3.33 MamdaFundamentalListener& Wombat::MamdaOptionContract::getFundamentalListener ()

Return the fundamental listener.

Returns:
The registered fundamental listener.

6.61.3.34 void Wombat::MamdaOptionContract::setInView (bool inView)

Set whether this contract is in the "view" within the option chain.
Parameters:

\textit{inView} Set whether this contract is in the "view" within the option chain.

See also:

MamdaOptionChain

6.61.3.35 \textbf{bool Wombat::MamdaOptionContract::getInView ()}

Return whether this contract is in the "view" within the option chain.

Returns:

Whether this contract is in the "view" within the option chain.

See also:

MamdaOptionChain

6.61.3.36 \textbf{int64_t Wombat::MamdaOptionContract::getSeqNum ()}

6.61.3.37 \textbf{void Wombat::MamdaOptionContract::setSeqNum (int64_t num)}

The documentation for this class was generated from the following file:

- MamdaOptionContract.h
A class that represents a set of option contracts at a given strike price.

#include <MamdaOptionContractSet.h>

Inheritance diagram for Wombat::MamdaOptionContractSet::

```
map< const char *, Wombat::MamdaOptionContract *, Wombat::char_str_less_than >
```

Public Member Functions

- **MamdaOptionContractSet ()**
  
  Set the contract for the best bid and offer.

- **~MamdaOptionContractSet ()**

- **void setBboContract (MamdaOptionContract *contract)**
  
  Set the contract for the best bid and offer.

- **void setWombatBboContract (MamdaOptionContract *contract)**
  
  Set the contract for the best bid and offer, as calculated by NYSE Technologies.

- **void setExchangeContract (const char *exchange, MamdaOptionContract *contract)**
  
  Set the contract for the particular exchange.

- **MamdaOptionContract * getBboContract () const**
  
  Return the contract for the best bid and offer.

- **MamdaOptionContract * getWombatBboContract () const**
  
  Return the contract for the best bid and offer, as calculated by NYSE Technologies.

- **MamdaOptionContract * getExchangeContract (const char *exchange) const**
  
  Return the contract for the particular exchange.

### 6.62.1 Detailed Description

A class that represents a set of option contracts at a given strike price.
6.62 Wombat::MamdaOptionContractSet Class Reference

6.62.2 Constructor & Destructor Documentation

6.62.2.1 Wombat::MamdaOptionContractSet::MamdaOptionContractSet()

6.62.2.2 Wombat::MamdaOptionContractSet::~MamdaOptionContractSet()

6.62.3 Member Function Documentation

6.62.3.1 void Wombat::MamdaOptionContractSet::setBboContract (MamdaOptionContract * contract)

Set the contract for the best bid and offer.

6.62.3.2 void Wombat::MamdaOptionContractSet::setWombatBboContract (MamdaOptionContract * contract)

Set the contract for the best bid and offer, as calculated by NYSE Technologies.

6.62.3.3 void Wombat::MamdaOptionContractSet::setExchangeContract (const char * exchange, MamdaOptionContract * contract)

Set the contract for the particular exchange.

6.62.3.4 MamdaOptionContract* Wombat::MamdaOptionContractSet::getBboContract () const

Return the contract for the best bid and offer.

6.62.3.5 MamdaOptionContract* Wombat::MamdaOptionContractSet::getWombatBboContract () const

Return the contract for the best bid and offer, as calculated by NYSE Technologies.

6.62.3.6 MamdaOptionContract* Wombat::MamdaOptionContractSet::getExchangeContract (const char * exchange) const

Return the contract for the particular exchange.

The documentation for this class was generated from the following file:

- MamdaOptionContractSet.h
6.63 Wombat::MamdaOptionExchangeUtils Class Reference

A class with static utility functions for dealing with exchanges.

#include <MamdaOptionExchangeUtils.h>

Static Public Member Functions

- static bool isBbo (const char *exchange)
  
  Return whether the exchange ID is the one used to represent the best bid and offer.
  
  Currently it will match "", "BBO" and "Z".

  Parameters:

  exchange  The exchange Id.

  Returns:

  Whether the given exchange id is being used to identify the BBO (Best Bid and Offer) record.

- static bool isWombatBbo (const char *exchange)
  
  Return whether the exchange ID is the one used to represent the NYSE Technologies-calculated best bid and offer.

6.63.1 Detailed Description

A class with static utility functions for dealing with exchanges.

6.63.2 Member Function Documentation

6.63.2.1 static bool Wombat::MamdaOptionExchangeUtils::isBbo (const char * exchange)  [static]

Return whether the exchange ID is the one used to represent the best bid and offer.
Currently it will match "", "BBO" and "Z".

Parameters:

  exchange  The exchange Id.

Returns:

  Whether the given exchange id is being used to identify the BBO (Best Bid and Offer) record.

6.63.2.2 static bool Wombat::MamdaOptionExchangeUtils::isWombatBbo (const char * exchange)  [static]

Return whether the exchange ID is the one used to represent the NYSE Technologies-calculated best bid and offer.
Currently will match "BBO",

**Parameters:**

- `exchange` The exchange id.

**Returns:**

Whether the given exchange id is being used to identify the WombatBBO (Best Bid and Offer) record.

The documentation for this class was generated from the following file:

- [MamdaOptionExchangeUtils.h](#)
Wombat::MamdaOptionExpirationDateSet Class Reference

A class that represents a set of expiration dates, each of which contains a set of strike prices, each of which contains a set of option contracts, each of which contains exchange-specific contracts.

#include <MamdaOptionExpirationDateSet.h>

Inheritance diagram for Wombat::MamdaOptionExpirationDateSet::

```
    map

Wombat::MamdaOptionExpirationDateSet
```

Public Member Functions

- MamdaOptionExpirationDateSet()
- MamdaOptionExpirationDateSet(const MamdaOptionExpirationDateSet &copy)
- ~MamdaOptionExpirationDateSet()
- MamdaOptionExpirationDateSet & operator=(const MamdaOptionExpirationDateSet &rhs)
- void getExpirationsBefore(MamdaOptionExpirationDateSet &result, const MamaDateTime &date) const
- void getExpirations(MamdaOptionExpirationDateSet &result, int numExpirations) const
- const char * toString() const

6.64.1 Detailed Description

A class that represents a set of expiration dates, each of which contains a set of strike prices, each of which contains a set of option contracts, each of which contains exchange-specific contracts.

To access a set of strike prices for a given expiration date, use the find method (inherited from std::map).
6.64 Wombat::MamdaOptionExpirationDateSet Class Reference

6.64.2 Constructor & Destructor Documentation

6.64.2.1 Wombat::MamdaOptionExpirationDateSet::MamdaOptionExpirationDateSet ()

6.64.2.2 Wombat::MamdaOptionExpirationDateSet::MamdaOptionExpirationDateSet (const MamdaOptionExpirationDateSet & copy)

6.64.2.3 Wombat::MamdaOptionExpirationDateSet::~MamdaOptionExpirationDateSet ()

6.64.3 Member Function Documentation

6.64.3.1 MamdaOptionExpirationDateSet& Wombat::MamdaOptionExpirationDateSet::operator= (const MamdaOptionExpirationDateSet & rhs)

6.64.3.2 void Wombat::MamdaOptionExpirationDateSet::getExpirationsBefore (MamdaOptionExpirationDateSet & result, const MamaDateTime & date) const

6.64.3.3 void Wombat::MamdaOptionExpirationDateSet::getExpirations (MamdaOptionExpirationDateSet & result, int numExpirations) const

6.64.3.4 const char* Wombat::MamdaOptionExpirationDateSet::toString () const

The documentation for this class was generated from the following file:

- MamdaOptionExpirationDateSet.h
6.65 Wombat::MamdaOptionExpirationStrikes Class Reference

A class that represents a set of strike prices at a particular expiration date.

#include <MamdaOptionExpirationStrikes.h>

Inheritance diagram for Wombat::MamdaOptionExpirationStrikes::

```
map
Wombat::MamdaOptionExpirationStrikes
```

Public Member Functions

- **MamdaOptionExpirationStrikes ()**
- **MamdaOptionExpirationStrikes (const MamdaOptionExpirationStrikes &copy)**
- **~MamdaOptionExpirationStrikes ()**
- **void trimStrikes (const StrikeSet &strikeSet)**

*Trim the current set of strike prices to the given set.*

6.65.1 Detailed Description

A class that represents a set of strike prices at a particular expiration date.

Each strike price of which contains a set of option contracts, each of which contains exchange-specific contracts. To access a contract set for a given strike price, use the get method (inherited from TreeMap).
6.65 Wombat::MamdaOptionExpirationStrikes Class Reference

6.65.2 Constructor & Destructor Documentation

6.65.2.1 Wombat::MamdaOptionExpirationStrikes::MamdaOptionExpirationStrikes()

6.65.2.2 Wombat::MamdaOptionExpirationStrikes::MamdaOptionExpirationStrikes(const MamdaOptionExpirationStrikes & copy)

6.65.2.3 Wombat::MamdaOptionExpirationStrikes::~MamdaOptionExpirationStrikes()

6.65.3 Member Function Documentation

6.65.3.1 void Wombat::MamdaOptionExpirationStrikes::trimStrikes(const StrikeSet & strikeSet)

Trim the current set of strike prices to the given set.

Parameters:

strikeSet The set of strikes to trim the current set to.

The documentation for this class was generated from the following file:

- MamdaOptionExpirationStrikes.h
6.66 Wombat::MamdaOptionFields Class Reference

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing options related fields from update messages.

```cpp
#include <MamdaOptionFields.h>
```

Inheritance diagram for Wombat::MamdaOptionFields:

```
Wombat::MamdaOptionFields
Wombat::MamdaFields
```

### Static Public Member Functions

- static void `setDictionary` (const MamaDictionary &dictionary)
- static void `reset` ()
- static bool `isSet` ()

### Static Public Attributes

- static const MamaFieldDescriptor * `CONTRACT_SYMBOL`
- static const MamaFieldDescriptor * `UNDERLYING_SYMBOL`
- static const MamaFieldDescriptor * `EXPIRATION_DATE`
- static const MamaFieldDescriptor * `STRIKE_PRICE`
- static const MamaFieldDescriptor * `PUT_CALL`
- static const MamaFieldDescriptor * `OPEN_INTEREST`
- static const MamaFieldDescriptor * `EXERCISE_STYLE`

### 6.66.1 Detailed Description

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing options related fields from update messages.

This class should be initialized prior to using the MamdaOptionChainListener by calling `setDictionary()` with a valid dictionary object which contains options related fields.
6.66.2  Member Function Documentation

6.66.2.1 static void Wombat::MamdaOptionFields::setDictionary (const MamaDictionary & dictionary)  [static]

6.66.2.2 static void Wombat::MamdaOptionFields::reset ()  [static]

6.66.2.3 static bool Wombat::MamdaOptionFields::isSet ()  [static]

6.66.3  Member Data Documentation

6.66.3.1 const MamaFieldDescriptor* Wombat::MamdaOptionFields::CONTRACT_SYMBOL  [static]

6.66.3.2 const MamaFieldDescriptor* Wombat::MamdaOptionFields::UNDERLYING_SYMBOL  [static]

6.66.3.3 const MamaFieldDescriptor* Wombat::MamdaOptionFields::EXPIRATION_DATE  [static]

6.66.3.4 const MamaFieldDescriptor* Wombat::MamdaOptionFields::STRIKE_PRICE  [static]

6.66.3.5 const MamaFieldDescriptor* Wombat::MamdaOptionFields::PUT_CALL  [static]

6.66.3.6 const MamaFieldDescriptor* Wombat::MamdaOptionFields::OPEN_INTEREST  [static]

6.66.3.7 const MamaFieldDescriptor* Wombat::MamdaOptionFields::EXERCISE_STYLE  [static]

The documentation for this class was generated from the following file:

- MamdaOptionFields.h
MamdaOptionSeriesUpdate is an interface that provides access to fields related to option series update events.

#include <MamdaOptionSeriesUpdate.h>

Inheritance diagram for Wombat::MamdaOptionSeriesUpdate::

```
Wombat::MamdaBasicEvent
    Wombat::MamdaOptionSeriesUpdate
        Wombat::MamdaOptionChainListener
```

**Public Member Functions**

- `const MamdaOptionContract *getOptionContract () const`
- `char getOptionAction () const`

  *Return the action related to the last series update message.*

**Static Public Attributes**

- static const char `ACTION_UNKNOWN = '"'
- static const char `ACTION_ADD = 'A'
- static const char `ACTION_DELETE = 'D'

**6.67.1 Detailed Description**

MamdaOptionSeriesUpdate is an interface that provides access to fields related to option series update events.

Update events include adds/removes of contracts to the chain.
6.67.2 Member Function Documentation

6.67.2.1 const MamdaOptionContract∗ Wombat::MamdaOptionSeriesUpdate::getOptionContract () const

Returns:
The option contract to which the most recent event applies. If the contract is new, it will have already been added to the chain. If it is being removed, it will have already been removed from the chain.

Reimplemented in Wombat::MamdaOptionChainListener.

6.67.2.2 char Wombat::MamdaOptionSeriesUpdate::getOptionAction () const

Return the action related to the last series update message.

Returns:
The action related to the last series update message.

Reimplemented in Wombat::MamdaOptionChainListener.

6.67.3 Member Data Documentation

6.67.3.1 const char Wombat::MamdaOptionSeriesUpdate::ACTION_ UNKNOWN = ’ ’ [static]

6.67.3.2 const char Wombat::MamdaOptionSeriesUpdate::ACTION_ADD = ’A’ [static]

6.67.3.3 const char Wombat::MamdaOptionSeriesUpdate::ACTION_DELETE = ’D’ [static]

The documentation for this class was generated from the following file:

- MamdaOptionSeriesUpdate.h
A class that represents the call and put contract sets at a given strike price.

#include <MamdaOptionStrikeSet.h>

Public Member Functions

- **MamdaOptionStrikeSet** (const MamaDateTime &expireDate, double strikePrice)
- **~MamdaOptionStrikeSet** ()
- **MamdaOptionContractSet * getCallSet** ()
  
  Return a set of call contracts at the given strike price.

- **const MamdaOptionContractSet * getCallSet** () const
  
  Return a set of call contracts at the given strike price.

- **MamdaOptionContractSet * getPutSet** ()
  
  Return a set of put contracts at the given strike price.

- **const MamdaOptionContractSet * getPutSet** () const
  
  Return a set of put contracts at the given strike price.

- **const MamaDateTime & getExpireDate** () const
  
  Return the expiration date for the contracts at this strike price.

- **const char * getExpireDateStr** () const
  
  Return the expiration date for the contracts at this strike price as a string.

- **double getStrikePrice** () const
  
  Return the strike price.

Detailed Description

A class that represents the call and put contract sets at a given strike price.
6.68.2 Constructor & Destructor Documentation

6.68.2.1 Wombat::MamdaOptionStrikeSet::MamdaOptionStrikeSet (const MamaDateTime & expireDate, double strikePrice)

6.68.2.2 Wombat::MamdaOptionStrikeSet::~MamdaOptionStrikeSet ()

6.68.3 Member Function Documentation

6.68.3.1 MamdaOptionContractSet* Wombat::MamdaOptionStrikeSet::getCallSet ()
Return a set of call contracts at the given strike price.

Returns:
    All call contracts at the given strike price.

6.68.3.2 const MamdaOptionContractSet* Wombat::MamdaOptionStrikeSet::getCallSet () const
Return a set of call contracts at the given strike price.

Returns:
    All call contracts at the given strike price. (const pointer)

6.68.3.3 MamdaOptionContractSet* Wombat::MamdaOptionStrikeSet::getPutSet ()
Return a set of put contracts at the given strike price.

Returns:
    All put contracts at the given strike price.

6.68.3.4 const MamdaOptionContractSet* Wombat::MamdaOptionStrikeSet::getPutSet () const
Return a set of put contracts at the given strike price.

Returns:
    All put contracts at the given strike price. (const pointer)
6.68.3.5 const MamaDateTime& Wombat::MamdaOptionStrikeSet::getExpireDate () const

Return the expiration date for the contracts at this strike price.

Returns:
The expiration date for the contracts at this strike price.

6.68.3.6 const char∗ Wombat::MamdaOptionStrikeSet::getExpireDateStr () const

Return the expiration date for the contracts at this strike price as a string.

Returns:
The expiration date for the contracts at this strike price.

6.68.3.7 double Wombat::MamdaOptionStrikeSet::getStrikePrice () const

Return the strike price.

Returns:
The strike price represented by this object.

The documentation for this class was generated from the following file:

- MamdaOptionStrikeSet.h
MamdaOrderBook is a class that provides order book functionality, including iterators over price levels and entries within price levels.

```cpp
#include <MamdaOrderBook.h>
```

### Public Types

- `typedef const bidIterator constBidIterator`
- `typedef const askIterator constAskIterator`
- `typedef const bidEntryIterator constBidEntryIterator`
- `typedef const askEntryIterator constAskEntryIterator`

### Public Member Functions

- `MamdaOrderBook ()`
- `~MamdaOrderBook ()`
- `MamdaOrderBook (const MamdaOrderBook &)`
- `MamdaOrderBook & operator= (const MamdaOrderBook &)`
- `void clear (bool deleteLevels=true)`
  
  Clear the order book entirely.

- `void setSymbol (const char *symbol)`
  
  The order book subscription symbol.

- `const char * getSymbol () const`
  
  The orderbook subscription symbol.

- `void setPartId (const char *partId)`
  
  The orderbook participant id.

- `const char * getPartId () const`
  
  The orderbook participant id.

- `bool hasPartId () const`
  
  The orderbook participant id.

- `MamdaOrderBookPriceLevel * findOrCreateLevel (double price, MamdaOrderBookPriceLevel::Side side)`
  
  Create a price level in the orderbook for the given price/size.
• MamdaOrderBookPriceLevel * findOrCreateLevel (MamaPrice &price, MamdaOrderBookPriceLevel::Side side)
• MamdaOrderBookPriceLevel * findOrCreateLevel (double price, MamdaOrderBookPriceLevel::Side side, MamdaOrderBookPriceLevel::Action &action)
• MamdaOrderBookPriceLevel * findOrCreateLevel (MamaPrice &price, MamdaOrderBookPriceLevel::Side side, MamdaOrderBookPriceLevel::Action &action)
• MamdaOrderBookPriceLevel * findLevel (double price, MamdaOrderBookPriceLevel::Side side)
• MamdaOrderBookPriceLevel * findLevel (MamaPrice &price, MamdaOrderBookPriceLevel::Side side)
• void addLevel (const MamdaOrderBookPriceLevel &level)
  Add a price level to the orderbook.
• void updateLevel (const MamdaOrderBookPriceLevel &level)
  Update an existing level in the orderbook.
• void deleteLevel (const MamdaOrderBookPriceLevel &level)
  Delete a price level from the orderbook.
• void apply (const MamdaOrderBook &deltaBook)
  Apply a delta to this (presumably) full book.
• void apply (const MamdaOrderBookBasicDelta &delta)
  Apply a delta to this book.
• void apply (const MamdaOrderBookBasicDeltaList &delta)
  Apply a delta to this book.
• void applyMarketOrder (const MamdaOrderBookBasicDelta &delta)
  Apply a market order delta to this book.
• void applyMarketOrder (const MamdaOrderBookBasicDeltaList &delta)
  Apply a market order delta to this book.
• void copy (const MamdaOrderBook &rhs)
  Copy a book.
• void setAsDeltaDeleted (const MamdaOrderBook &bookToDelete)
  Set this order book to be a delta that would, when applied, delete all of the fields in the bookToDelete.
• void setAsDeltaDifference (const MamdaOrderBook &lhs, const MamdaOrderBook &rhs)
  Set this order book to be a delta that would, when applied, be the difference between
two other books.

• size_t getTotalNumLevels () const
  Get the total number of price levels (both sides of order book).

• size_t getNumBidLevels () const
  Get the number of bid price levels.

• size_t getNumAskLevels () const
  Get the number of ask price levels.

• MamdaOrderBookPriceLevel * getBidMarketOrders () const
  Get the bid market orders.

• MamdaOrderBookPriceLevel * getAskMarketOrders () const
  Get the ask market orders.

• MamdaOrderBookPriceLevel * getMarketOrdersSide (MamdaOrderBookPriceLevel::Side side)
  Get the market orders for the specified side.

• MamdaOrderBookPriceLevel * getOrCreateMarketOrdersSide (MamdaOrderBookPriceLevel::Side side)
  Get the market orders for the specified side.

• void detach (MamdaOrderBookPriceLevel *level)
  Detach the given level from the book.

• void detach (MamdaOrderBookEntry *entry)
  Add the given entry to the detach list to be cleaned up.

• void cleanupDetached ()
  Free resources associated with any detached price levels or entries, detached either
through explicit calls to detach() or detached as a result or having no remaining
entries.

• const MamaDateTime & getBookTime () const
  Get the "book time" (or "event time") of the last update.

• void setBookTime (const MamaDateTime &bookTime) const
Set the BookTime for this order book.

- void setSourceDerivative (const MamaSourceDerivative *sourceDeriv)
  
  *Set the MamaSourceDerivative for this order book.*

- const MamaSourceDerivative * getSourceDerivative () const
  
  *Get the MamaSourceDerivative for this order book.*

- const MamaSource * getSource () const
  
  *Get the MamaSource for this order book.*

- void setQuality (mamaQuality quality)
  
  *Set the mamaQuality for this order book.*

- mamaQuality getQuality () const
  
  *Get the mamaQuality for this order book.*

- void setClosure (void *closure)
  
  *Set the order book closure handle.*

- void * getClosure () const
  
  *Get the order book closure handle.*

- bool operator==(const MamdaOrderBook &rhs) const
  
  *Equality operator.*

- bool operator!=(const MamdaOrderBook &rhs) const
  
  *Non-equality operator.*

- void addEntry (MamdaOrderBookEntry *entry, double price, MamdaOrderBookPriceLevel::Side side, const MamaDateTime &eventTime, MamdaOrderBookBasicDelta *delta)
  
  *Add an entry to the order book and (if “delta” is not NULL) record information about the delta related to this action.*

- void addEntry (MamdaOrderBookEntry *entry, MamaPrice &price, MamdaOrderBookPriceLevel::Side side, const MamaDateTime &eventTime, MamdaOrderBookBasicDelta *delta)
  
  *Add an entry to the order book and (if “delta” is not NULL) record information about the delta related to this action.*
• MamdaOrderBookEntry * addEntry (const char *entryId, mama_quantity_t entrySize, double price, MamdaOrderBookPriceLevel::Side side, const MamaDateTime &eventTime, const MamaSourceDerivative *sourceDeriv, MamdaOrderBookBasicDelta *delta)

  Add an entry to the order book and (if "delta" is not NULL) record information about the delta related to this action.

• MamdaOrderBookEntry * addEntry (const char *entryId, mama_quantity_t entrySize, MamaPrice &price, MamdaOrderBookPriceLevel::Side side, const MamaDateTime &eventTime, const MamaSourceDerivative *sourceDeriv, MamdaOrderBookBasicDelta *delta)

  Add an entry to the order book and (if "delta" is not NULL) record information about the delta related to this action.

• void updateEntry (MamdaOrderBookEntry *entry, mama_quantity_t size, const MamaDateTime &eventTime, MamdaOrderBookBasicDelta *delta)

  Update an entry in the order book and (if "delta" is not NULL) record information about the delta related to this action.

• void deleteEntry (MamdaOrderBookEntry *entry, const MamaDateTime &eventTime, MamdaOrderBookBasicDelta *delta)

  Delete an entry in the order book and (if "delta" is not NULL) record information about the delta related to this action.

• void addEntriesFromBook (const MamdaOrderBook *book, MamdaOrderBookEntryFilter *filter, MamdaOrderBookBasicDeltaList *delta)

  Add all entries from another book into this book.

• void addPriceLevelsFromBookAsEntries (const MamdaOrderBook *book, const char *source, MamdaOrderBookBasicDeltaList *delta)

  Add all price levels from another book as entries (one per price level) into this book using "source" as the entryId for each entry.

• void deleteEntriesFromSource (const MamaSource *source, MamdaOrderBookBasicDeltaList *delta)

  Delete all entries in this book that have "source" as its MamaSource.

• bool reevaluate ()

  Re-evaluate the order book.

• void setNeedsReevaluation (bool need)

  Set whether this book needs a re-evaluation.

• bool getNeedsReevaluation () const
Get whether this book needs a re-evaluation.

- void setCheckSourceState (bool check)
  
  Set whether to check the MamaSourceState when adding/deleting/re-evaluating entries in the book.

- bool getCheckSourceState () const
  
  Get whether to check the MamaSourceState when adding/deleting/re-evaluating entries in the book.

- MamdaOrderBookPriceLevel * getLevelAtPrice (double price, MamdaOrderBookPriceLevel::Side side) const
  
  Return the order book price level at "price" on "side" of the order book.

- MamdaOrderBookPriceLevel * getLevelAtPosition (mama_u32_t pos, MamdaOrderBookPriceLevel::Side side) const
  
  Return the order book price level at position "pos" in the order book.

- MamdaOrderBookEntry * getEntryAtPosition (mama_u32_t pos, MamdaOrderBookPriceLevel::Side side) const
  
  Return the order book entry at position "pos" in the order book.

- void assertEqual (const MamdaOrderBook &rhs) const
  
  Order book equality verification.

- bidIterator bidBegin ()
- constBidIterator bidBegin () const
- bidIterator bidEnd ()
- constBidIterator bidEnd () const
- askIterator askBegin ()
- constAskIterator askBegin () const
- askIterator askEnd ()
- constAskIterator askEnd () const
- bidEntryIterator bidEntryBegin ()
- constBidEntryIterator bidEntryBegin () const
- bidEntryIterator bidEntryEnd ()
- constBidEntryIterator bidEntryEnd () const
- askEntryIterator askEntryBegin ()
- constAskEntryIterator askEntryBegin () const
- askEntryIterator askEntryEnd ()
- constAskEntryIterator askEntryEnd () const
- void setIsConsistent (bool isConsistent)
  
  Set whether the order book is in a consistent or an inconsistent state.
• bool getIsConsistent () const
  Get whether the order book is in a consistent or an inconsistent state.

• void dump (ostream &output) const
  Dump the order book to the output stream.

• void generateDeltaMsgs (bool generate)
  Enable the generation of book deltas for this book.

• bool getGenerateDeltaMsgs ()
  Get whether book delta generation is enabled.

• bool populateDelta (MamaMsg &msg)
  Populate a MamaMsg of the changes to this order book.

• void populateRecap (MamaMsg &msg)
  Populate a MamaMsg with the current state of this order book.

• void addDelta (MamdaOrderBookEntry *entry, MamdaOrderBookPriceLevel *level, mama_quantity_t plDeltaSize, MamdaOrderBookPriceLevel::Action plAction, MamdaOrderBookEntry::Action entAction)
  For book delta generation.

• void clearDeltaList ()
  Clear the delta list using for storing generated deltas

• void setBookContributors (const char *book Contributors)
  Set the orderbook contributors.

• const char * getBookContributors () const
  Get the orderbook contributors.

• bool hasBookContributors () const
  Is the orderbook contributors set.

• bool getBookContributorsModified () const
  Get whether the books contributors have been modified.

• void setBookContributorsModified (bool modified)
  Set whether the books contributors have been modified.
Static Public Member Functions

- static void setStrictChecking (bool strict)

Enforce strict checking of order book modifications (at the expense of some performance).

Classes

- class askEntryIterator
- class askIterator
- class bidEntryIterator
- class bidIterator

6.69.1 Detailed Description

MamdaOrderBook is a class that provides order book functionality, including iterators over price levels and entries within price levels.
6.69.2 Member Typedef Documentation

6.69.2.1 typedef const bidIterator Wombat::MamdaOrderBook::constBidIterator

6.69.2.2 typedef const askIterator Wombat::MamdaOrderBook::constAskIterator

6.69.2.3 typedef const bidEntryIterator Wombat::MamdaOrderBook::constBidEntryIterator

6.69.2.4 typedef const askEntryIterator Wombat::MamdaOrderBook::constAskEntryIterator

6.69.3 Constructor & Destructor Documentation

6.69.3.1 Wombat::MamdaOrderBook::MamdaOrderBook ()

6.69.3.2 Wombat::MamdaOrderBook::~MamdaOrderBook ()

6.69.3.3 Wombat::MamdaOrderBook::MamdaOrderBook (const MamdaOrderBook &)

6.69.4 Member Function Documentation

6.69.4.1 MamdaOrderBook& Wombat::MamdaOrderBook::operator= (const MamdaOrderBook &)

6.69.4.2 void Wombat::MamdaOrderBook::clear (bool deleteLevels = true)

Clear the order book entirely.

Parameters:

deleteLevels  If true then the level objects will be deleted

6.69.4.3 void Wombat::MamdaOrderBook::setSymbol (const char * symbol)

The order book subscription symbol.

Parameters:

symbol  The subscription symbol.
6.69.4.4 const char* Wombat::MamdaOrderBook::getSymbol (const char* const) const

The orderbook subscription symbol.

**Returns:**

The orderbook subscription symbol.

6.69.4.5 void Wombat::MamdaOrderBook::setPartId (const char* partId)

The orderbook participant id.

**Parameters:**

- **partId** The participant id

6.69.4.6 const char* Wombat::MamdaOrderBook::getPartId (const char* const) const

The orderbook participant id.

**Returns:**

The orderbook participant id

6.69.4.7 bool Wombat::MamdaOrderBook::hasPartId () const

The orderbook participant id.

**Returns:**

Whether this orderbook has a participant id

6.69.4.8 MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::findOrCreateLevel (double price, MamdaOrderBookPriceLevel::Side side)

Create a price level in the orderbook for the given price/size.

The price level is initially empty and marked as "not used". The "not used" status changes automatically when entries are added to the price level.
Parameters:

- **price**  The price of the price level to find/create.
- **side**  The side of the book of the price level to find/create.

Returns:

- The found or newly create price level.

Exceptions:

- `<MamdaOrderBookException>`  When an error is encountered during book processing.

6.69.4.9  **MamdaOrderBookPriceLevel**  `Wombat::MamdaOrderBook::findOrCreateLevel (MamaPrice & price, MamdaOrderBookPriceLevel::Side side)`

6.69.4.10  **MamdaOrderBookPriceLevel**  `Wombat::MamdaOrderBook::findOrCreateLevel (double price, MamdaOrderBookPriceLevel::Side side, MamdaOrderBookPriceLevel::Action & action)`

6.69.4.11  **MamdaOrderBookPriceLevel**  `Wombat::MamdaOrderBook::findOrCreateLevel (MamaPrice & price, MamdaOrderBookPriceLevel::Side side, MamdaOrderBookPriceLevel::Action & action)`

6.69.4.12  **MamdaOrderBookPriceLevel**  `Wombat::MamdaOrderBook::findLevel (double price, MamdaOrderBookPriceLevel::Side side)`

6.69.4.13  **MamdaOrderBookPriceLevel**  `Wombat::MamdaOrderBook::findLevel (MamaPrice & price, MamdaOrderBookPriceLevel::Side side)`

6.69.4.14  **void**  `Wombat::MamdaOrderBook::addLevel (const MamdaOrderBookPriceLevel & level)`

Add a price level to the orderbook.

Parameters:

- **level**  The price level to add to the orderbook.

Exceptions:

- `<MamdaOrderBookException>`  When an error is encountered during book processing.

---

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.69.4.15  void Wombat::MamdaOrderBook::updateLevel (const MamdaOrderBookPriceLevel & level)

Update an existing level in the orderbook.

Parameters:

level  The details of the price level to update.

Exceptions:

<MamdaOrderBookException>  When an error is encountered during book processing.

6.69.4.16  void Wombat::MamdaOrderBook::deleteLevel (const MamdaOrderBookPriceLevel & level)

Delete a price level from the orderbook.

Parameters:

level  The price level to delete from the orderbook.

Exceptions:

<MamdaOrderBookException>  When an error is encountered during book processing.

6.69.4.17  void Wombat::MamdaOrderBook::apply (const MamdaOrderBook & deltaBook)

Apply a delta to this (presumably) full book.

Parameters:

deltaBook  The delta to apply to the orderbook.

Exceptions:

<MamdaOrderBookException>  When an error is encountered during book processing.
Apply a delta to this book.

Parameters:

\textit{delta}  The simple delta to apply to the orderbook.

Exceptions:

\textit{MamdaOrderBookException}  When an error is encountered during book processing.

Apply a delta to this book.

Parameters:

\textit{delta}  The complex delta to apply to the orderbook.

Exceptions:

\textit{MamdaOrderBookException}  When an error is encountered during book processing.

Apply a market order delta to this book.

Parameters:

\textit{delta}  The simple market order delta to apply to the orderbook.

Exceptions:

\textit{MamdaOrderBookException}  When an error is encountered during book processing.
6.69.4.21  void Wombat::MamdaOrderBook::applyMarketOrder (const MamdaOrderBookBasicDeltaList & delta)

Apply a market order delta to this book.

**Parameters:**

*delta*  The market order complex delta to apply to the orderbook.

**Exceptions:**

*<MamdaOrderBookException>*  When an error is encountered during book processing.

6.69.4.22  void Wombat::MamdaOrderBook::copy (const MamdaOrderBook & rhs)

Copy a book.

**Parameters:**

*rhs*  The orderbook to copy.

**Exceptions:**

*<MamdaOrderBookException>*  When an error is encountered during book processing.

6.69.4.23  void Wombat::MamdaOrderBook::setAsDeltaDeleted (const MamdaOrderBook & bookToDelete)

Set this order book to be a delta that would, when applied, delete all of the fields in the bookToDelete.

6.69.4.24  void Wombat::MamdaOrderBook::setAsDeltaDifference (const MamdaOrderBook & lhs, const MamdaOrderBook & rhs)

Set this order book to be a delta that would, when applied, be the difference between two other books.

6.69.4.25  size_t Wombat::MamdaOrderBook::getTotalNumLevels () const

Get the total number of price levels (both sides of order book).
6.69 Wombat::MamdaOrderBook Class Reference

Returns:
   The total number of levels in the book.

6.69.4.26 size_t Wombat::MamdaOrderBook::getNumBidLevels () const

Get the number of bid price levels.

Returns:
   The total number of bid levels in the book.

6.69.4.27 size_t Wombat::MamdaOrderBook::getNumAskLevels () const

Get the number of ask price levels.

Returns:
   The total number of ask levels in the book.

6.69.4.28 MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::getBidMarketOrders () const

Get the bid market orders.

Returns:
   The market order bid level.

6.69.4.29 MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::getAskMarketOrders () const

Get the ask market orders.

Returns:
   The market order ask level.
6.69.4.30 MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::getMarketOrdersSide (MamdaOrderBookPriceLevel::Side side)

Get the market orders for the specified side.
Will return NULL if no market orders exist in the book.

Parameters:

- **side** The market order side the price level to get.

Returns:

The market order ask level.

6.69.4.31 MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::getOrCreateMarketOrdersSide (MamdaOrderBookPriceLevel::Side side)

Get the market orders for the specified side.
Will create an empty level if none exist.

Parameters:

- **side** The market order side the price level to get.

Returns:

The market order ask level.

6.69.4.32 void Wombat::MamdaOrderBook::detach (MamdaOrderBookPriceLevel * level)

Detach the given level from the book.

Parameters:

- **level** The level to detach

6.69.4.33 void Wombat::MamdaOrderBook::detach (MamdaOrderBookEntry * entry)

Add the given entry to the detach list to be cleaned up.
Parameters:

entry  The entry to detach

6.69.4.34  void Wombat::MamdaOrderBook::cleanupDetached ()

Free resources associated with any detached price levels or entries, detached either through explicit calls to detach() or detached as a result or having no remaining entries.

6.69.4.35  const MamaDateTime& Wombat::MamdaOrderBook::getBookTime () const

Get the "book time" (or "event time") of the last update.
The book time is related to market data feeds: the time that the market data feed suggests that the update actually happened.

Returns:

The book time of the last update.

6.69.4.36  void Wombat::MamdaOrderBook::setBookTime (const MamaDateTime & bookTime) const

Set the BookTime for this order book.

Parameters:

bookTime  The book time

6.69.4.37  void Wombat::MamdaOrderBook::setSourceDerivative (const MamaSourceDerivative * sourceDeriv)

Set the MamaSourceDerivative for this order book.

Parameters:

sourceDeriv  The source derivative
6.69.4.38 const MamaSourceDerivative* Wombat::MamdaOrderBook::getSourceDerivative () const

Get the MamaSourceDerivative for this order book.

**Returns:**
- The source derivative.

6.69.4.39 const MamaSource* Wombat::MamdaOrderBook::getSource () const

Get the MamaSource for this order book.

**Returns:**
- The source.

6.69.4.40 void Wombat::MamdaOrderBook::setQuality (mamaQuality quality)

Set the mamaQuality for this order book.

**Parameters:**
- *quality* The new quality.

6.69.4.41 mamaQuality Wombat::MamdaOrderBook::getQuality () const

Get the mamaQuality for this order book.

**Returns:**
- The quality.

6.69.4.42 void Wombat::MamdaOrderBook::setClosure (void * closure)

Set the order book closure handle.

**Parameters:**
- *closure* The closure.
6.69.4.43  void* Wombat::MamdaOrderBook::getClosure () const

Get the order book closure handle.

Returns:

The orderbook closure.

6.69.4.44  bool Wombat::MamdaOrderBook::operator== (const MamdaOrderBook & rhs) const

Equality operator.
Two books are equal if their symbols, price levels and price level entries are identical.

Parameters:

rhs  The book to compare this book to.

Returns:

Whether the two books are equal.

6.69.4.45  bool Wombat::MamdaOrderBook::operator!= (const MamdaOrderBook & rhs) const

Non-equality operator.
Two books are equal if their symbols, price levels and price level entries are identical.

Parameters:

rhs  The book to compare this book to.

Returns:

Whether the two books are not equal.

399  { return ! operator== (rhs); }  

6.69.4.46  void Wombat::MamdaOrderBook::addEntry (MamdaOrderBookEntry * entry, double price, MamdaOrderBookPriceLevel::Side side, const MamaDateTime & eventTime, MamdaOrderBookBasicDelta * delta)

Add an entry to the order book and (if "delta" is not NULL) record information about the delta related to this action.
6.69.4.47 void Wombat::MamdaOrderBook::addEntry
(MamdaOrderBookEntry * entry, MamaPrice & price,
MamdaOrderBookPriceLevel::Side side, const MamaDateTime &
eventTime, MamdaOrderBookBasicDelta * delta)

Add an entry to the order book and (if "delta" is not NULL) record information about
the delta related to this action.

6.69.4.48 MamdaOrderBookEntry* Wombat::MamdaOrderBook::addEntry
(const char * entryId, mama_quantity_t entrySize, double price,
MamdaOrderBookPriceLevel::Side side, const MamaDateTime &
eventTime, const MamaSourceDerivative * sourceDeriv,
MamdaOrderBookBasicDelta * delta)

Add an entry to the order book and (if "delta" is not NULL) record information about
the delta related to this action.
The new entry is returned.

6.69.4.49 MamdaOrderBookEntry* Wombat::MamdaOrderBook::addEntry
(const char * entryId, mama_quantity_t entrySize, MamaPrice &
price, MamdaOrderBookPriceLevel::Side side, const MamaDateTime &
eventTime, const MamaSourceDerivative * sourceDeriv,
MamdaOrderBookBasicDelta * delta)

Add an entry to the order book and (if "delta" is not NULL) record information about
the delta related to this action.
The new entry is returned.

6.69.4.50 void Wombat::MamdaOrderBook::updateEntry
(MamdaOrderBookEntry * entry, mama_quantity_t size, const
MamaDateTime & eventTime, MamdaOrderBookBasicDelta * delta)

Update an entry in the order book and (if "delta" is not NULL) record information
about the delta related to this action.
If the entry is not internally "wired" to the order book, a MamdaOrderBookInvalidEntry
exception is thrown.
6.69.4.51 void Wombat::MamdaOrderBook::deleteEntry
(MamdaOrderBookEntry * entry, const MamaDateTime &
eventTime, MamdaOrderBookBasicDelta * delta)

Delete an entry in the order book and (if "delta" is not NULL) record information about
the delta related to this action.

If the entry is not internally "wired" to the order book, a MamdaOrderBookInvalidEntry
exception is thrown.

6.69.4.52 void Wombat::MamdaOrderBook::addEntriesFromBook (const
MamdaOrderBook * book, MamdaOrderBookEntryFilter * filter,
MamdaOrderBookBasicDeltaList * delta)

Add all entries from another book into this book.

Parameters:

book  The source book to add.
filter  If not NULL, a filter to apply to each entry.
delta  An optional delta to collect the added entries.

6.69.4.53 void Wombat::MamdaOrderBook::addPriceLevelsFromBookAs-
Entries (const MamdaOrderBook * book, const char * source,
MamdaOrderBookBasicDeltaList * delta)

Add all price levels from another book as entries (one per price level) into this book
using "source" as the entryId for each entry.

Parameters:

book  The source book to add.
source  The name to use as the entry ID.
delta  An optional delta to collect the added entries.

6.69.4.54 void Wombat::MamdaOrderBook::deleteEntriesFromSource (const
MamaSource * source, MamdaOrderBookBasicDeltaList * delta)

Delete all entries in this book that have "source" as its MamaSource.

Parameters:

source  The source to match.
delta  An optional delta to collect the deleted entries.
6.69.4.55  bool Wombat::MamdaOrderBook::reevaluate ()

Re-evaluate the order book.

This would be performed after the status of sources and/or subsources of an "aggregated order book" (i.e. a book built from multiple sources) have changed.

**Returns:**

Whether the book info changed based on the re-evaluation.

6.69.4.56  void Wombat::MamdaOrderBook::setNeedsReevaluation (bool need)

Set whether this book needs a re-evaluation.

6.69.4.57  bool Wombat::MamdaOrderBook::getNeedsReevaluation () const

Get whether this book needs a re-evaluation.

6.69.4.58  void Wombat::MamdaOrderBook::setCheckSourceState (bool check)

Set whether to check the MamaSourceState when adding/deleting/re-evaluating entries in the book.

6.69.4.59  bool Wombat::MamdaOrderBook::getCheckSourceState () const

Get whether to check the MamaSourceState when adding/deleting/re-evaluating entries in the book.

6.69.4.60  MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::getLevelAtPrice (double price, MamdaOrderBookPriceLevel::Side side) const

Return the order book price level at "price" on "side" of the order book.

**Parameters:**

- **price**  The price of the order book price level.
- **side**  The side of the order book to search.

**Returns:**

The order book price level or NULL if not found.
6.69.4.61 MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::getLevelAtPosition (mama_u32_t pos, MamdaOrderBookPriceLevel::Side side) const

Return the order book price level at position "pos" in the order book.

Parameters:

    pos    The position of the order book price level.
    side   The side of the order book to search.

Returns:

    The order book price level or NULL if not found.

6.69.4.62 MamdaOrderBookEntry* Wombat::MamdaOrderBook::getEntryAtPosition (mama_u32_t pos, MamdaOrderBookPriceLevel::Side side) const

Return the order book entry at position "pos" in the order book.

Parameters:

    pos    The position of the order book entry.
    side   The side of the order book to search

Returns:

    The order book entry or NULL if not found.

6.69.4.63 void Wombat::MamdaOrderBook::assertEqual (const MamdaOrderBook & rhs) const

Order book equality verification.

A MamdaOrderBookException is thrown if the books are not equal, along with the reason for the inequality.

Parameters:

    rhs    The book to compare this book to.

Exceptions:

    MamdaOrderBookException   When an error is encountered during book process-
Set whether the order book is in a consistent or an inconsistent state.

This method is typically called from within the `MamdaOrderBookListener` in response to sequence number gap detection and subsequent recovery from a gap event.

**Parameters:**

- `isConsistent` Whether the book is in a consistent state.
6.69.4.81 bool Wombat::MamdaOrderBook::getIsConsistent () const

Get whether the order book is in a consistent or an an inconsistent state.

A book is marked as being inconsistent by the MamdaOrderBookListener whenever a sequence number gap in the book updates is detected. The order book will be marked as consistent once again once a recap for the book is received by the MamdaOrderBookListener.

Returns:

Whether the book is in a consistent state.

6.69.4.82 void Wombat::MamdaOrderBook::dump (ostream & output) const

Dump the order book to the output stream.

Parameters:

output The ostream to write the orderbook to.

6.69.4.83 static void Wombat::MamdaOrderBook::setStrictChecking (bool strict) [static]

Enforce strict checking of order book modifications (at the expense of some performance).

This setting is passed on to the MamdaOrderBookPriceLevel and MamdaOrderBookEntry classes.

6.69.4.84 void Wombat::MamdaOrderBook::generateDeltaMsgs (bool generate)

Enable the generation of book deltas for this book.

When delta generation is enabled changes to the book are saved and can be popultaed to MamaMsgs.

Parameters:

publish Whether book delta generation is enabled.
6.69.4.85 bool Wombat::MamdaOrderBook::getGenerateDeltaMsgs ()

Get whether book delta generation is enabled.

Returns:

Whether book delta generation is enabled.

6.69.4.86 bool Wombat::MamdaOrderBook::populateDelta (MamaMsg & msg)

Populate a MamaMsg of the changes to this order book.

This will include the changes from the last time this function was called or all changes from the initial state.

Parameters:

msg     A MamaMsg ref containing all changes to the current book.

6.69.4.87 void Wombat::MamdaOrderBook::populateRecap (MamaMsg & msg)

Populate a MamaMsg with the current state of this order book.

Parameters:

msg     A MamaMsg containing all book, price and entry (if applicable) details of the current book.

6.69.4.88 void Wombat::MamdaOrderBook::addDelta (MamdaOrderBookEntry * entry, MamdaOrderBookPriceLevel * level, mama_quantity_t plDeltaSize, MamdaOrderBookPriceLevel::Action plAction, MamdaOrderBookEntry::Action entAction)

For book delta generation.

Add a delta to the order book delta list for the publishing of order book data.

Parameters:

tenry  MamdaOrderBookEntry where change occurred.
level   MamdaOrderBookPriceLevel where change occurred.
plDeltaSize  Pricelevel size change.
plAction  Pricelevel action.
entAction  Entry action.
6.69 Wombat::MamdaOrderBook Class Reference 363

6.69.4.89  void Wombat::MamdaOrderBook::clearDeltaList ()
clear the delta list using for storing generated deltas

6.69.4.90  void Wombat::MamdaOrderBook::setBookContributors (const char ∗ bookContributors)
Set the orderbook contributors.

Parameters:

bookContributors  The orderbook contributors

6.69.4.91  const char ∗ Wombat::MamdaOrderBook::getBookContributors () const
Get the orderbook contributors.

Returns:

The orderbook contributors

6.69.4.92  bool Wombat::MamdaOrderBook::hasBookContributors () const
Is the orderbook contributors set.

Returns:

Whether this orderbook has a list of contributors

6.69.4.93  bool Wombat::MamdaOrderBook::getBookContributorsModified () const
Get whether the books contributors have been modified.

Returns:

Whether the book contributors have been modified.
Set whether the books contributors have been modified.

**Parameters:**

- **modifies**  Whether the book contributors have been modified.

The documentation for this class was generated from the following file:

- MamdaOrderBook.h
#include <MamdaOrderBook.h>

## Public Member Functions

- askEntryIterator()
- askEntryIterator(const askEntryIterator &copy)
- askEntryIterator(const askEntryIteratorImpl &copy)
- ~askEntryIterator()
- askEntryIterator & operator=(const askEntryIterator &rhs)
- askEntryIterator & operator++()
- const askEntryIterator & operator++() const
- bool operator==(const askEntryIterator &rhs) const
- bool operator!=(const askEntryIterator &rhs) const
- MamdaOrderBookEntry * operator * ()
- const MamdaOrderBookEntry * operator * () const

## Protected Attributes

- askEntryIteratorImpl & mImpl

## Friends

- class MamdaOrderBook
6.70 Wombat::MamdaOrderBook::askEntryIterator Class Reference

6.70.1 Constructor & Destructor Documentation

6.70.1.1 Wombat::MamdaOrderBook::askEntryIterator::askEntryIterator()

6.70.1.2 Wombat::MamdaOrderBook::askEntryIterator::askEntryIterator
(const askEntryIterator & copy)

6.70.1.3 Wombat::MamdaOrderBook::askEntryIterator::askEntryIterator
(const askEntryIteratorImpl & copy)

6.70.1.4 Wombat::MamdaOrderBook::askEntryIterator::~askEntryIterator()

6.70.2 Member Function Documentation

6.70.2.1 askEntryIterator& Wombat::MamdaOrderBook::askEntryIterator::operator= (const askEntryIterator & rhs)

6.70.2.2 askEntryIterator& Wombat::MamdaOrderBook::askEntryIterator::operator++()

6.70.2.3 const askEntryIterator & Wombat::MamdaOrderBook::askEntryIterator::operator++() const

6.70.2.4 bool Wombat::MamdaOrderBook::askEntryIterator::operator==(const askEntryIterator & rhs) const

6.70.2.5 bool Wombat::MamdaOrderBook::askEntryIterator::operator!=(const askEntryIterator & rhs) const

6.70.2.6 MamdaOrderBookEntry* Wombat::MamdaOrderBook::askEntryIterator::operator*() const

6.70.2.7 const MamdaOrderBookEntry* Wombat::MamdaOrderBook::askEntryIterator::operator*() const

6.70.3 Friends And Related Function Documentation

6.70.3.1 friend class MamdaOrderBook [friend]

6.70.4 Member Data Documentation

6.70.4.1 askEntryIteratorImpl& Wombat::MamdaOrderBook::askEntryIterator::mImpl [protected]
- MamdaOrderBook.h
#include <MamdaOrderBook.h>

Public Member Functions

- askIterator()
- askIterator(const askIterator& copy)
- askIterator(const askIteratorImpl& copy)
- ~askIterator()
- askIterator & operator=(const askIterator &rhs)
- askIterator & operator++ ()
- const askIterator & operator++ () const
- askIterator & operator-- ()
- const askIterator & operator-- () const
- bool operator==(const askIterator &rhs) const
- bool operator!=(const askIterator &rhs) const
- MamdaOrderBookPriceLevel * operator* ()
- const MamdaOrderBookPriceLevel * operator* () const

Protected Attributes

- askIteratorImpl & mImpl

Friends

- class MamdaOrderBook
6.71 Wombat::MamdaOrderBook::askIterator Class Reference

6.71.1 Constructor & Destructor Documentation

6.71.1.1 Wombat::MamdaOrderBook::askIterator::askIterator ()

6.71.1.2 Wombat::MamdaOrderBook::askIterator::askIterator (const askIterator & copy)

6.71.1.3 Wombat::MamdaOrderBook::askIterator::askIterator (const askIteratorImpl & copy)

6.71.1.4 Wombat::MamdaOrderBook::askIterator::~askIterator ()

6.71.2 Member Function Documentation

6.71.2.1 askIterator& Wombat::MamdaOrderBook::askIterator::operator= (const askIterator & rhs)

6.71.2.2 askIterator& Wombat::MamdaOrderBook::askIterator::operator++ ()

6.71.2.3 const askIterator& Wombat::MamdaOrderBook::askIterator::operator++ () const

6.71.2.4 askIterator& Wombat::MamdaOrderBook::askIterator::operator– ()

6.71.2.5 const askIterator& Wombat::MamdaOrderBook::askIterator::operator– () const

6.71.2.6 bool Wombat::MamdaOrderBook::askIterator::operator== (const askIterator & rhs) const

6.71.2.7 bool Wombat::MamdaOrderBook::askIterator::operator!= (const askIterator & rhs) const

6.71.2.8 MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::askIterator::operator* ()

6.71.2.9 const MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::askIterator::operator* () const

6.71.3 Friends And Related Function Documentation

6.71.3.1 friend class MamdaOrderBook [friend]

6.71.4 Member Data Documentation

6.71.4.1 askIteratorImpl& Wombat::MamdaOrderBook::askIterator::mImpl [protected]

The documentation for this class was generated from the following file:
• MamdaOrderBook.h
#include <MamdaOrderBook.h>

**Public Member Functions**

- `bidEntryIterator ()`
- `bidEntryIterator (const bidEntryIterator &copy)`
- `bidEntryIterator (const bidEntryIteratorImpl &copy)`
- `~bidEntryIterator ()`
- `bidEntryIterator & operator= (const bidEntryIterator &rhs)`
- `bidEntryIterator & operator++ ()`
- `const bidEntryIterator & operator++ () const`
- `bool operator== (const bidEntryIterator &rhs)`
- `bool operator!= (const bidEntryIterator &rhs)` const
- `MamdaOrderBookEntry * operator * ()`
- `const MamdaOrderBookEntry * operator * () const`

**Protected Attributes**

- `bidEntryIteratorImpl & mImpl`

**Friends**

- `class MamdaOrderBook`
6.72.1 Constructor & Destructor Documentation

6.72.1.1 Wombat::MamdaOrderBook::bidEntryIterator::bidEntryIterator()

6.72.1.2 Wombat::MamdaOrderBook::bidEntryIterator::bidEntryIterator(const bidEntryIterator &copy)

6.72.1.3 Wombat::MamdaOrderBook::bidEntryIterator::bidEntryIterator(const bidEntryIteratorImpl &copy)

6.72.1.4 Wombat::MamdaOrderBook::bidEntryIterator::~bidEntryIterator()

6.72.2 Member Function Documentation

6.72.2.1 bidEntryIterator& Wombat::MamdaOrderBook::bidEntryIterator::operator= (const bidEntryIterator &rhs)

6.72.2.2 bidEntryIterator& Wombat::MamdaOrderBook::bidEntryIterator::operator++ ()

6.72.2.3 const bidEntryIterator& Wombat::MamdaOrderBook::bidEntryIterator::operator++ () const

6.72.2.4 bool Wombat::MamdaOrderBook::bidEntryIterator::operator== (const bidEntryIterator &rhs) const

6.72.2.5 bool Wombat::MamdaOrderBook::bidEntryIterator::operator!= (const bidEntryIterator &rhs) const

6.72.2.6 MamdaOrderBookEntry* Wombat::MamdaOrderBook::bidEntryIterator::operator* ()

6.72.2.7 const MamdaOrderBookEntry* Wombat::MamdaOrderBook::bidEntryIterator::operator* () const

6.72.3 Friends And Related Function Documentation

6.72.3.1 friend class MamdaOrderBook [friend]

6.72.4 Member Data Documentation

6.72.4.1 bidEntryIteratorImpl & Wombat::MamdaOrderBook::bidEntryIterator::mImpl [protected]
• MamdaOrderBook.h
#include <MamdaOrderBook.h>

Public Member Functions

- `bidIterator ()`
- `bidIterator (const bidIterator &copy)`
- `bidIterator (const bidIteratorImpl &copy)`
- `~bidIterator ()`
- `bidIterator & operator= (const bidIterator &rhs)`
- `bidIterator & operator++ ()`
- `const bidIterator & operator++ () const`
- `bidIterator & operator– ()`
- `const bidIterator & operator– () const`
- `bool operator== (const bidIterator &rhs) const`
- `bool operator!= (const bidIterator &rhs) const`
- `MamdaOrderBookPriceLevel * operator * ()`
- `const MamdaOrderBookPriceLevel * operator * () const`

Protected Attributes

- `bidIteratorImpl & mImpl`

Friends

- `class MamdaOrderBook`
6.73 Wombat::MamdaOrderBook::bidIterator Class Reference

6.73.1 Constructor & Destructor Documentation

6.73.1.1 Wombat::MamdaOrderBook::bidIterator::bidIterator ()

6.73.1.2 Wombat::MamdaOrderBook::bidIterator::bidIterator (const bidIterator & copy)

6.73.1.3 Wombat::MamdaOrderBook::bidIterator::bidIterator (const bidIteratorImpl & copy)

6.73.1.4 Wombat::MamdaOrderBook::bidIterator::~bidIterator ()

6.73.2 Member Function Documentation

6.73.2.1 bidIterator& Wombat::MamdaOrderBook::bidIterator::operator= (const bidIterator & rhs)

6.73.2.2 bidIterator& Wombat::MamdaOrderBook::bidIterator::operator++ ()

6.73.2.3 const bidIterator& Wombat::MamdaOrderBook::bidIterator::operator++ () const

6.73.2.4 bidIterator& Wombat::MamdaOrderBook::bidIterator::operator– ()

6.73.2.5 const bidIterator& Wombat::MamdaOrderBook::bidIterator::operator– () const

6.73.2.6 bool Wombat::MamdaOrderBook::bidIterator::operator== (const bidIterator & rhs) const

6.73.2.7 bool Wombat::MamdaOrderBook::bidIterator::operator!= (const bidIterator & rhs) const

6.73.2.8 MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::bidIterator::operator* ()

6.73.2.9 const MamdaOrderBookPriceLevel* Wombat::MamdaOrderBook::bidIterator::operator* () const

6.73.3 Friends And Related Function Documentation

6.73.3.1 friend class MamdaOrderBook [friend]

6.73.4 Member Data Documentation

6.73.4.1 bidIteratorImpl& Wombat::MamdaOrderBook::bidIterator::mImpl
• MamdaOrderBook.h
MamdaOrderBookBasicDelta is a class that saves information about a basic order book delta.

```cpp
#include <MamdaOrderBookBasicDelta.h>
```

Inheritance diagram for Wombat::MamdaOrderBookBasicDelta::

```
Wombat::MamdaOrderBookBasicDelta
|      |
|      | Wombat::MamdaOrderBookSimpleDelta
|      | Wombat::MamdaOrderBookConcreteSimpleDelta
```

### Public Member Functions

- `MamdaOrderBookBasicDelta ()`
- `MamdaOrderBookBasicDelta (const MamdaOrderBookBasicDelta &)`
- `virtual ~MamdaOrderBookBasicDelta ()`
- `virtual void clear ()`
  
  *Clear the delta.*

- `virtual void set (MamdaOrderBookEntry *entry, MamdaOrderBookPriceLevel *level, mama_quantity_t plDeltaSize, MamdaOrderBookPriceLevel::Action plAction, MamdaOrderBookEntry::Action entryAction)`
  
  *Set the delta info.*

- `void setPriceLevel (MamdaOrderBookPriceLevel *level)`
  
  *Set the MamdaOrderBookPriceLevel object to which this entry belongs.*

- `void setPIDeltaAction (MamdaOrderBookPriceLevel::Action action)`
  
  *Set the delta action with respect to the price level.*

- `void setPIDeltaSize (mama_quantity_t size)`
  
  *Set the price level delta size.*

- `void applyPIDeltaSize (mama_quantity_t size)`
  
  *Apply the price level delta size to the existing size, giving the net effect.*
• MamdaOrderBookPriceLevel * getPriceLevel () const
  Get the MamdaOrderBookPriceLevel object related to this basic delta.

• MamdaOrderBookEntry * getEntry () const
  Get the MamdaOrderBookEntry object related to this basic delta.

• mama_quantity_t getPlDeltaSize () const
  Get the difference in size for the price level.

• MamdaOrderBookPriceLevel::Action getPlDeltaAction () const
  Get the delta action with respect to the price level.

• MamdaOrderBookEntry::Action getEntryDeltaAction () const
  Get the delta action with respect to the entry.

• void setEntryDeltaAction (MamdaOrderBookEntry::Action action)
  Get the delta action with respect to the entry.

• virtual const MamdaOrderBook * getOrderByBook () const
  Get the MamdaOrderBook object to which this delta belongs.

• void dump (ostream &output) const
  Dump the simple update to the output stream.

Protected Attributes

• MamdaOrderBookPriceLevel * mPriceLevel
• MamdaOrderBookEntry * mEntry
• mama_quantity_t mPlDeltaSize
• MamdaOrderBookPriceLevel::Action mPlAction
• MamdaOrderBookEntry::Action mEntryAction

6.74.1 Detailed Description

MamdaOrderBookBasicDelta is a class that saves information about a basic order book delta.
A basic delta is one that affects a single order book entry.
6.74 Wombat::MamdaOrderBookBasicDelta Class Reference

6.74.2 Constructor & Destructor Documentation

6.74.2.1 Wombat::MamdaOrderBookBasicDelta::MamdaOrderBookBasicDelta ()

49    { clear(); }  

6.74.2.2 Wombat::MamdaOrderBookBasicDelta::MamdaOrderBookBasicDelta (const MamdaOrderBookBasicDelta &)

6.74.2.3 virtual Wombat::MamdaOrderBookBasicDelta::~MamdaOrderBookBasicDelta () [virtual]

51    {}  

6.74.3 Member Function Documentation

6.74.3.1 virtual void Wombat::MamdaOrderBookBasicDelta::clear ()
[virtual]

Clear the delta.

6.74.3.2 virtual void Wombat::MamdaOrderBookBasicDelta::set (MamdaOrderBookEntry * entry, MamdaOrderBookPriceLevel * level,
mama_quantity_t plDeltaSize, MamdaOrderBookPriceLevel::Action plaAction, MamdaOrderBookEntry::Action entryAction) [virtual]

Set the delta info.

6.74.3.3 void Wombat::MamdaOrderBookBasicDelta::setPriceLevel
(MamdaOrderBookPriceLevel * level)

Set the MamdaOrderBookPriceLevel object to which this entry belongs.
This method is invoked internally, by the MAMDA API, when an entry is added to a price level.

Parameters:

level The price level to be associated with.

75    { mPriceLevel = level; }
6.74.3.4 void Wombat::MamdaOrderBookBasicDelta::setPlDeltaAction
(MamdaOrderBookPriceLevel::Action action)

Set the delta action with respect to the price level.

Parameters:

  action The price level action

83 { mPlAction = action; }

6.74.3.5 void Wombat::MamdaOrderBookBasicDelta::setPlDeltaSize
(mama_quantity_t size)

Set the price level delta size.

Parameters:

  action The price level action

91 { mPlDeltaSize = size; }

6.74.3.6 void Wombat::MamdaOrderBookBasicDelta::applyPlDeltaSize
(mama_quantity_t size)

Apply the price level delta size to the existing size, giving the net effect.

Parameters:

  action The price level action

101 { mPlDeltaSize += size; }

6.74.3.7 MamdaOrderBookPriceLevel* Wombat::MamdaOrderBookBasicDelta::getPriceLevel () const

Get the MamdaOrderBookPriceLevel object related to this basic delta.

Returns:

  The price level.

109 { return mPriceLevel; }
6.74.8 MamdaOrderBookEntry* Wombat::MamdaOrderBookBasicDelta::getEntry() const

Get the MamdaOrderBookEntry object related to this basic delta.

Returns:
The entry.

117 { return mEntry; }

6.74.9 mama_quantity_t Wombat::MamdaOrderBookBasicDelta::getPlDeltaSize() const

Get the difference in size for the price level.

Returns:
The price level size delta.

125 { return mPlDeltaSize; }

6.74.10 MamdaOrderBookPriceLevel::Action Wombat::MamdaOrderBookBasicDelta::getPlDeltaAction() const

Get the delta action with respect to the price level.

Returns:
The price level action.

133 { return mPlAction; }

6.74.11 MamdaOrderBookEntry::Action Wombat::MamdaOrderBookBasicDelta::getEntryDeltaAction() const

Get the delta action with respect to the entry.

Returns:
The entry action.

141 { return mEntryAction; }

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.74.3.12  void Wombat::MamdaOrderBookBasicDelta::setEntryDeltaAction (MamdaOrderBookEntry::Action action)

Get the delta action with respect to the entry.

```cpp
148  { mEntryAction = action; }
```

6.74.3.13  virtual const MamdaOrderBook* Wombat::MamdaOrderBookBasicDelta::getOrderBook () const [virtual]

Get the MamdaOrderBook object to which this delta belongs.

Returns:

The order book related to this delta.

6.74.3.14  void Wombat::MamdaOrderBookBasicDelta::dump (ostream & output) const

Dump the simple update to the output stream.

Parameters:

`output`  The `ostream` to write the update to.

6.74.4  Member Data Documentation

6.74.4.1  MamdaOrderBookPriceLevel* Wombat::MamdaOrderBookBasicDelta::mPriceLevel  [protected]

6.74.4.2  MamdaOrderBookEntry* Wombat::MamdaOrderBookBasicDelta::mEntry  [protected]

6.74.4.3  mama_quantity_t Wombat::MamdaOrderBookBasicDelta::mPlDeltaSize  [protected]

6.74.4.4  MamdaOrderBookPriceLevel::Action Wombat::MamdaOrderBookBasicDelta::mPlAction  [protected]

6.74.4.5  MamdaOrderBookEntry::Action Wombat::MamdaOrderBookBasicDelta::mEntryAction  [protected]

The documentation for this class was generated from the following file:
- MamdaOrderBookBasicDelta.h
MamdaOrderBookBasicDeltaList is a class that saves information about an order book delta that involves multiple entries and/or price levels.

```cpp
#include <MamdaOrderBookBasicDeltaList.h>
```

Inheritance diagram for Wombat::MamdaOrderBookBasicDeltaList::

```
Wombat::MamdaOrderBookBasicDeltaList
Wombat::MamdaOrderBookComplexDelta
Wombat::MamdaOrderBookConcreteComplexDelta
```

### Public Types

- typedef const iterator const_iterator
- MOD_SIDES_NONE = 0
- MOD_SIDES_BID = 1
- MOD_SIDES_ASK = 2
- MOD_SIDES_BID_AND_ASK = 3
- enum ModifiedSides { MOD_SIDES_NONE = 0, MOD_SIDES_BID = 1, MOD_SIDES_ASK = 2, MOD_SIDES_BID_AND_ASK = 3 }

### Public Member Functions

- MamdaOrderBookBasicDeltaList ()
- ~MamdaOrderBookBasicDeltaList ()
- void clear ()
  
  *Clear the delta.*

- void setKeepBasicDeltas (bool keep)
  
  *Set whether to actually keep the basic deltas.*

- ModifiedSides getModifiedSides () const
  
  *Get which side(s) of the book have been modified by this complex update.*
• void add (MamdaOrderBookEntry *entry, MamdaOrderBookPriceLevel *level, mama_quantity_t plDeltaSize, MamdaOrderBookPriceLevel::Action plAction, MamdaOrderBookEntry::Action entryAction)
  * Add a basic delta.

• void add (const MamdaOrderBookBasicDelta &delta)
  * Add a basic delta.

• void setOrderBook (MamdaOrderBook *book)
  * Set the MamdaOrderBook object to which this delta belongs.

• MamdaOrderBook * getOrderBook () const
  * Get the MamdaOrderBook object to which this delta belongs.

• mama_size_t getSize () const
  * Return the number of simple deltas in this complex delta.

• void fixPriceLevelActions ()
  * Fix up price level actions (temporary workaround for problem).

• void setConflateDeltas (bool conflate)
  * Whether to conflate the order book deltas.

• bool getSendImmediately ()
  * Get sendImmediately.

• void setProcessEntries (bool processEntries)
  * Set whether we are interested in "entry level" information at all.

• void dump (ostream &output) const
  * Dump the complex update to the output stream.

• iterator begin ()
• const_iterator begin () const
• iterator end ()
• const_iterator end () const

Classes

• class iterator
  * The MamdaOrderBookBasicDeltaList's iterator provides access to the list of MamdaOrderBookBasicDelta objects that comprise it.
6.75.1 Detailed Description

MamdaOrderBookBasicDeltaList is a class that saves information about an order book delta that involves multiple entries and/or price levels.

For example, a modified order may involve a price change that means moving an entry from one price level to another. A delta list is made up of several basic deltas, which can be iterated over by methods provided in the class.

6.75.2 Member Typedef Documentation

6.75.2.1 typedef const iterator Wombat::MamdaOrderBookBasicDeltaList::const_iterator

6.75.3 Member Enumeration Documentation

6.75.3.1 enum Wombat::MamdaOrderBookBasicDeltaList::ModifiedSides

Enumerator:

MOD_SIDES_NONE
MOD_SIDES_BID
MOD_SIDES_ASK
MOD_SIDES_BID_AND_ASK

{  MOD_SIDES_NONE = 0,
  MOD_SIDES_BID = 1,
  MOD_SIDES_ASK = 2,
  MOD_SIDES_BID_AND_ASK = 3
};

6.75.4 Constructor & Destructor Documentation

6.75.4.1 Wombat::MamdaOrderBookBasicDeltaList::MamdaOrderBookBasicDeltaList ()

6.75.4.2 Wombat::MamdaOrderBookBasicDeltaList::~MamdaOrderBookBasicDeltaList ()

6.75.5 Member Function Documentation

6.75.5.1 void Wombat::MamdaOrderBookBasicDeltaList::clear ()

Clear the delta.
6.75.5.2  void Wombat::MamdaOrderBookBasicDeltaList::setKeepBasicDeltas(bool keep)

Set whether to actually keep the basic deltas.
Many applications don’t need the basic deltas and will iterate over part or all of the
full book (with the deltas already applied). If this is set to true and an attempt is made
to iterate over the basic deltas (by calling begin() or end()) then a MamdaOrderBook-
Exception will be thrown.

6.75.5.3 ModifiedSides Wombat::MamdaOrderBookBasicDeltaList::get-
ModifiedSides() const

Get which side(s) of the book have been modified by this complex update.
This information may prevent the need for receivers of complex updates to iterate over
one or other side of the book.

Returns:
The modified side(s).

6.75.5.4 void Wombat::MamdaOrderBookBasicDeltaList::add(Mamda-
OrderBookEntry* entry, MamdaOrderBookPriceLevel* level,
mama_quantity_t plDeltaSize, MamdaOrderBookPriceLevel::Action
plAction, MamdaOrderBookEntry::Action entryAction)

Add a basic delta.
This method adds a MamdaOrderBookBasicDelta to the list.

6.75.5.5 void Wombat::MamdaOrderBookBasicDeltaList::add(const
MamdaOrderBookBasicDelta& delta)

Add a basic delta.
This method adds a copy of the MamdaOrderBookBasicDelta to the list.

6.75.5.6 void Wombat::MamdaOrderBookBasicDeltaList::setOrderBook
(MamdaOrderBook* book)

Set the MamdaOrderBook object to which this delta belongs.
Parameters:

book The order book related to this delta.
6.75.5.7 *MamdaOrderBook* Wombat::MamdaOrderBookBasicDeltaList::getOrderBook () const

Get the `MamdaOrderBook` object to which this delta belongs.

**Returns:**

The order book related to this delta.

6.75.5.8 mama_size_t Wombat::MamdaOrderBookBasicDeltaList::getSize () const

Return the number of simple deltas in this complex delta.

6.75.5.9 void Wombat::MamdaOrderBookBasicDeltaList::fixPriceLevelActions ()

Fix up price level actions (temporary workaround for problem).
This method ensures that all basic deltas for the same price level end up with the same
price level action.

6.75.5.10 void Wombat::MamdaOrderBookBasicDeltaList::setConflateDeltas (bool conflate)

Whether to conflate the order book deltas.

**Parameters:**

`conflate` Whether to conflate order book deltas.

6.75.5.11 bool Wombat::MamdaOrderBookBasicDeltaList::getSendImmediately ()

Get sendImmediately.

6.75.5.12 void Wombat::MamdaOrderBookBasicDeltaList::setProcessEntries (bool processEntries)

Set whether we are interested in "entry level" information at all.

**Parameters:**

`process` Whether to process entries in books.
6.75.5.13  void Wombat::MamdaOrderBookBasicDeltaList::dump (ostream & output) const

Dump the complex update to the output stream.

Parameters:
  
  output  The ostream to write the update to.

6.75.5.14  iterator Wombat::MamdaOrderBookBasicDeltaList::begin ()

6.75.5.15  const_iterator Wombat::MamdaOrderBookBasicDeltaList::begin () const

6.75.5.16  iterator Wombat::MamdaOrderBookBasicDeltaList::end ()

6.75.5.17  const_iterator Wombat::MamdaOrderBookBasicDeltaList::end () const

The documentation for this class was generated from the following file:

  * MamdaOrderBookBasicDeltaList.h
6.76 Wombat::MamdaOrderBookBasicDeltaList::iterator Class Reference

The MamdaOrderBookBasicDeltaList’s iterator provides access to the list of MamdaOrderBookBasicDelta objects that comprise it.

#include <MamdaOrderBookBasicDeltaList.h>

Public Member Functions

- iterator ()
- iterator (const iterator &copy)
- iterator (const iteratorImpl &copy)
- iterator (iteratorImpl &copy)
- ~iterator ()
- iterator & operator= (const iterator &rhs)
- iterator & operator++ ()
- const iterator & operator++ () const
- bool operator== (const iterator &rhs) const
- bool operator!= (const iterator &rhs) const
- MamdaOrderBookBasicDelta * operator * ()
- const MamdaOrderBookBasicDelta * operator * () const

Protected Attributes

- iteratorImpl & mImpl

Friends

- class MamdaOrderBookBasicDeltaList

6.76.1 Detailed Description

The MamdaOrderBookBasicDeltaList’s iterator provides access to the list of MamdaOrderBookBasicDelta objects that comprise it.
6.76.2 Constructor & Destructor Documentation

6.76.2.1 Wombat::MamdaOrderBookBasicDeltaList::iterator::iterator ()

6.76.2.2 Wombat::MamdaOrderBookBasicDeltaList::iterator::iterator (const iterator & copy)

6.76.2.3 Wombat::MamdaOrderBookBasicDeltaList::iterator::iterator (const iteratorImpl & copy)

6.76.2.4 Wombat::MamdaOrderBookBasicDeltaList::iterator::iterator (iteratorImpl & copy)

6.76.2.5 Wombat::MamdaOrderBookBasicDeltaList::iterator::~iterator ()

6.76.3 Member Function Documentation

6.76.3.1 iterator& Wombat::MamdaOrderBookBasicDeltaList::iterator::operator= (const iterator & rhs)

6.76.3.2 iterator& Wombat::MamdaOrderBookBasicDeltaList::iterator::operator++ ()

6.76.3.3 const iterator& Wombat::MamdaOrderBookBasicDeltaList::iterator::operator++ () const

6.76.3.4 bool Wombat::MamdaOrderBookBasicDeltaList::iterator::operator== (const iterator & rhs) const

6.76.3.5 bool Wombat::MamdaOrderBookBasicDeltaList::iterator::operator!= (const iterator & rhs) const

6.76.3.6 MamdaOrderBookBasicDelta* Wombat::MamdaOrderBookBasicDeltaList::iterator::operator* ()

6.76.3.7 const MamdaOrderBookBasicDelta* Wombat::MamdaOrderBookBasicDeltaList::iterator::operator* () const

6.76.4 Friends And Related Function Documentation

6.76.4.1 friend class MamdaOrderBookBasicDeltaList [friend]

6.76.5 Member Data Documentation

6.76.5.1 iteratorImpl& Wombat::MamdaOrderBookBasicDeltaList::iterator::mImpl [protected]

The documentation for this class was generated from the following file:
• MamdaOrderBookBasicDeltaList.h
MamdaOrderBookChecker is a class that provides order book sanity checking by periodically requesting snapshots of the order book from the publisher and comparing that with an order book being maintained in real time.

```cpp
#include <MamdaOrderBookChecker.h>
```

### Public Member Functions

- **MamdaOrderBookChecker** (const MamdaOrderBook &realTimeBook, const MamdaSubscription &realTimeSubsc, MamdaOrderBookListener &realTimeListener, MamdaOrderBookCheckerHandler *handler, mama_f64_t intervalSeconds)
  
  Constructor for the order book checker.

- **MamdaOrderBookChecker** (MamaQueue *queue, MamdaOrderBookCheckerHandler *handler, MamaSource *source, const char *symbol, mama_f64_t intervalSeconds)

- **~MamdaOrderBookChecker** ()
  
  Destructor.

- **void checkSnapShotNow ()**
  
  Perform an ad hoc snapshot check now.

- **mama_u32_t getSuccessCount ()** const

- **mama_u32_t getInconclusiveCount ()** const
  
  Return the number of inconclusive counts.

- **mama_u32_t getFailureCount ()** const
  
  Return the number of failed checks.

### Detailed Description

*MamdaOrderBookChecker* is a class that provides order book sanity checking by periodically requesting snapshots of the order book from the publisher and comparing that with an order book being maintained in real time.

This class is purely for testing purposes, to test for possible configuration or programming errors in the order book publisher and in MAMDA order book management code.
The developer registers a handler that contains callbacks for successful, inconclusive and failure events. The developer also provides an interval representing the frequency of the snapshot checks. The first check will take place at some random point in time between zero and the interval.

### Constructor & Destructor Documentation

#### 6.77.2.1 Wombat::MamdaOrderBookChecker::MamdaOrderBookChecker

```cpp
Wombat::MamdaOrderBookChecker::MamdaOrderBookChecker
(const MamdaOrderBook & realTimeBook, const MamdaSubscription & realTimeSubsc,
 MamdaOrderBookListener & realTimeListener,
 MamdaOrderBookCheckerHandler * handler, mama_f64_t intervalSeconds)
```

Constructor for the order book checker.
The caller registers a callback for events related to the checker.

**Parameters:**

- `realTimeBook` The real time order book to compare against. This param is "const" because we are not supposed to change it, even though it will change underneath us.
- `realTimeSubsc` The `MamdaSubscription` associated with the `realTimeBook`. We need it to access the message-level sequence number. This param is "const" because we are not supposed to change it, even though it will change underneath us.
- `realTimeListener` The `MamdaOrderBookListener` associated with the `realTimeBook`.
- `handler` The user-provided callback handler.
- `intervalSeconds` The number of seconds between checks.

#### 6.77.2.2 Wombat::MamdaOrderBookChecker::MamdaOrderBookChecker

```cpp
Wombat::MamdaOrderBookChecker::MamdaOrderBookChecker
(MamaQueue * queue, MamdaOrderBookCheckerHandler * handler,
 MamaSource * source, const char * symbol, mama_f64_t intervalSeconds)
```

#### 6.77.2.3 Wombat::MamdaOrderBookChecker::~MamdaOrderBookChecker

```cpp
Wombat::MamdaOrderBookChecker::~MamdaOrderBookChecker
()
```

Destructor.
6.77.3  Member Function Documentation

6.77.3.1  void Wombat::MamdaOrderBookChecker::checkSnapShotNow ()

Perform an ad hoc snapshot check now.
This may be useful if the checking is to be performed by some external trigger event.

6.77.3.2  mama_u32_t Wombat::MamdaOrderBookChecker::getSuccessCount () const

Returns:
The number of successful checks.

6.77.3.3  mama_u32_t Wombat::MamdaOrderBookChecker::getInconclusiveCount () const

Return the number of inconclusive counts.
An attempt to check the order book may be inconclusive if the order book sequence numbers do not match up.

Returns:
The number of inconclusive checks.

6.77.3.4  mama_u32_t Wombat::MamdaOrderBookChecker::getFailureCount () const

Return the number of failed checks.
This should be zero, of course.

Returns:
The number of failed checks.

The documentation for this class was generated from the following file:

- MamdaOrderBookChecker.h
MamdaOrderBookCheckerHandler is an interface for applications that want to handle the results of the MamdaOrderBookChecker.

#include <MamdaOrderBookCheckerHandler.h>

Public Member Functions

- virtual void onSuccess (MamdaOrderBookCheckType checkType)=0
  Method invoked when a successful check is completed.

- virtual void onInconclusive (MamdaOrderBookCheckType checkType, const char *reason)=0
  Method invoked when check is completed inconclusively.

- virtual void onFailure (MamdaOrderBookCheckType checkType, const char *reason, const MamaMsg *msg, const MamdaOrderBook &realTimeBook, const MamdaOrderBook &checkBook)=0
  Method invoked when a failed check is completed.

- virtual ~MamdaOrderBookCheckerHandler ()

6.78.1 Detailed Description

MamdaOrderBookCheckerHandler is an interface for applications that want to handle the results of the MamdaOrderBookChecker. Callback interfaces are provided for correct and erroneous checks.

6.78.2 Constructor & Destructor Documentation

6.78.2.1 virtual Wombat::MamdaOrderBookCheckerHandler::~MamdaOrderBookCheckerHandler () [virtual]

{}
6.78.3 Member Function Documentation

6.78.3.1 virtual void Wombat::MamdaOrderBookCheckerHandler::onSuccess (MamdaOrderBookCheckType checkType) [pure virtual]

Method invoked when a successful check is completed.

6.78.3.2 virtual void Wombat::MamdaOrderBookCheckerHandler::onInconclusive (MamdaOrderBookCheckType checkType, const char * reason) [pure virtual]

Method invoked when check is completed inconclusively.
An attempt to check the order book may be inconclusive if the order book sequence numbers do not match up.

6.78.3.3 virtual void Wombat::MamdaOrderBookCheckerHandler::onFailure (MamdaOrderBookCheckType checkType, const char * reason, const MamaMsg * msg, const MamdaOrderBook & realTimeBook, const MamdaOrderBook & checkBook) [pure virtual]

Method invoked when a failed check is completed.
The message provided, if non-NULL, is the one received for the snapshot or delta, depending upon the value of checkType.
The documentation for this class was generated from the following file:

- MamdaOrderBookCheckerHandler.h
Wombat::MamdaOrderBookClear Class Reference

MamdaOrderBookClear is an interface that provides access to order book related fields.
#include <MamdaOrderBookClear.h>

Inheritance diagram for Wombat::MamdaOrderBookClear::

```
Wombat::MamdaBasicEvent
    └── Wombat::MamdaOrderBookClear
```

Public Member Functions

- virtual const MamdaOrderBook ∗ getOrderBook () const =0
  
  Returns the full orderbook related to this clear event.

- virtual ~MamdaOrderBookClear ()

6.79.1 Detailed Description

MamdaOrderBookClear is an interface that provides access to order book related fields.

6.79.2 Constructor & Destructor Documentation

6.79.2.1 virtual Wombat::MamdaOrderBookClear::~MamdaOrderBookClear () [virtual]

48 {

6.79.3 Member Function Documentation

6.79.3.1 virtual const MamdaOrderBook ∗ Wombat::MamdaOrderBookClear::getOrderBook () const [pure virtual]

Returns the full orderbook related to this clear event.
Returns:

The full order book.

The documentation for this class was generated from the following file:

- MamdaOrderBookClear.h
6.80 Wombat::MamdaOrderBookComplexDelta Class Reference

MamdaOrderBookComplexDelta is a class that saves information about a complex order book delta.

#include <MamdaOrderBookComplexDelta.h>

Inheritance diagram for Wombat::MamdaOrderBookComplexDelta::

```
Wombat::MamdaOrderBookComplexDelta
```

Public Member Functions

- virtual ~MamdaOrderBookComplexDelta ()

6.80.1 Detailed Description

MamdaOrderBookComplexDelta is a class that saves information about a complex order book delta.

A complex delta involves multiple entries and/or price levels. For example, a modified order may involve a price change that means moving an entry from one price level to another. A complex delta is made up of several simple deltas, which can be iterated over by methods provided in the class.

6.80.2 Constructor & Destructor Documentation

6.80.2.1 virtual Wombat::MamdaOrderBookComplexDelta::~MamdaOrderBookComplexDelta () [virtual]

```
45 {}
```

The documentation for this class was generated from the following file:

- MamdaOrderBookComplexDelta.h
6.81 Wombat::MamdaOrderBookConcreteComplexDelta Class Reference

MamdaOrderBookConcreteComplexDelta is a class that saves information about a complex order book delta.

#include <MamdaOrderBookConcreteComplexDelta.h>

Inheritance diagram for Wombat::MamdaOrderBookConcreteComplexDelta::

```
Wombat::MamdaOrderBookConcreteComplexDelta
Wombat::MamdaOrderBookComplexDelta
Wombat::MamdaOrderBookBasicDeltaList Wombat::MamdaBasicEvent
```

Public Member Functions

- MamdaOrderBookConcreteComplexDelta ()
- virtual ~MamdaOrderBookConcreteComplexDelta ()
- virtual const char * getSymbol () const
  
  Get the instruments string symbol.

- virtual const char * getPartId () const
  
  Get the participant identifier.

- virtual mama_seqnum_t getEventSeqNum () const
  
  Get the event sequence number.

- virtual const MamaDateTime & getEventTime () const
  
  Get the event time.

- virtual const MamaDateTime & getSrcTime () const
  
  Get the source time.

- virtual const MamaDateTime & getActivityTime () const
  
  Get the activity time.

- virtual const MamaDateTime & getLineTime () const
  
  Get the line time.
• virtual const MamaDateTime & getSendTime () const
  
  *Get the send time.*

• virtual const MamaMsgQual & getMsgQual () const
  
  *Get the message qualifier.*

• virtual MamdaFieldState getSymbolFieldState () const
  
  *Get the string symbol field state.*

• virtual MamdaFieldState getPartIdFieldState () const
  
  *Get the participant identifier field state.*

• virtual MamdaFieldState getEventSeqNumFieldState () const
  
  *Get the event sequence number field state.*

• virtual MamdaFieldState getEventTimeFieldState () const
  
  *Get the event time field state.*

• virtual MamdaFieldState getSrcTimeFieldState () const
  
  *Get the source time field state.*

• virtual MamdaFieldState getActivityTimeFieldState () const
  
  *Get the activity time field state.*

• virtual MamdaFieldState.getLineTimeFieldState () const
  
  *Get the line time of the update.*

• virtual MamdaFieldState getSendTimeFieldState () const
  
  *Get the send time field state.*

• virtual MamdaFieldState.getMsgQualFieldState () const
  
  *Get the message qualifier field state.*

• virtual void setSymbol (const char *value)

• virtual void setPartId (const char *value)

• virtual void setEventSeqNum (mama_seqnum_t value)

• virtual void setEventTime (const MamaDateTime &value)

• virtual void setSrcTime (const MamaDateTime &value)

• virtual void setActivityTime (const MamaDateTime &value)

• virtual void setLineTime (const MamaDateTime &value)

• virtual void setSize (const MamaDateTime &value)

• virtual void setCmd (const MamaMsgQual &value)
6.81.1 Detailed Description

*MamdaOrderBookConcreteComplexDelta* is a class that saves information about a complex order book delta.

A complex delta involves multiple entries and/or price levels. For example, a modified order may involve a price change that means moving an entry from one price level to another. A complex delta is made up of several simple deltas, which can be iterated over by methods provided in the class.

6.81.2 Constructor & Destructor Documentation

6.81.2.1 Wombat::MamdaOrderBookConcreteComplexDelta::MamdaOrderBookConcreteComplexDelta ()

6.81.2.2 virtual Wombat::MamdaOrderBookConcreteComplexDelta::~MamdaOrderBookConcreteComplexDelta () [virtual]

6.81.3 Member Function Documentation

6.81.3.1 virtual const char ∗ Wombat::MamdaOrderBookConcreteComplexDelta::getSymbol () const [virtual]

Get the instruments string symbol.

**Returns:**

Symbol. This is the "well-known" symbol for the security, including any symbology mapping performed by the publisher.

Implements *Wombat::MamdaBasicEvent*.

6.81.3.2 virtual const char ∗ Wombat::MamdaOrderBookConcreteComplexDelta::getPartId () const [virtual]

Get the participant identifier.

**Returns:**

Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).
6.81 Wombat::MamdaOrderBookConcreteComplexDelta Class Reference

Implements Wombat::MamdaBasicEvent.

6.81.3.3 virtual mama_seqnum_t Wombat::MamdaOrderBookConcreteComplexDelta::getEventSeqNum () const
 [virtual]

Get the event sequence number.

Returns:
Source sequence number. The exchange generated sequence number.

Implements Wombat::MamdaBasicEvent.

6.81.3.4 virtual const MamaDateTime& Wombat::MamdaOrderBookConcreteComplexDelta::getEventTime () const
 [virtual]

Get the event time.

Returns:
Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.81.3.5 virtual const MamaDateTime& Wombat::MamdaOrderBookConcreteComplexDelta::getSrcTime () const
 [virtual]

Get the source time.

Returns:
Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.
6.81.3.6 virtual const MamaDateTime& Wombat::MamdaOrderBookConcreteComplexDelta::getActivityTime () const
    [virtual]

Get the activity time.

**Returns:**

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements Wombat::MamdaBasicEvent.

6.81.3.7 virtual const MamaDateTime& Wombat::MamdaOrderBookConcreteComplexDelta::getLineTime () const
    [virtual]

Get the line time.

**Returns:**

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

6.81.3.8 virtual const MamaDateTime& Wombat::MamdaOrderBookConcreteComplexDelta::getSendTime () const
    [virtual]

Get the send time.

**Returns:**

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime().

Implements Wombat::MamdaBasicEvent.
6.81 Wombat::MamdaOrderBookConcreteComplexDelta Class Reference 411

6.81.3.9 virtual const MamaMsgQual& Wombat::MamdaOrderBookConcreteComplexDelta::getMsgQual () const
    [virtual]

Get the message qualifier.

Returns:
    Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.81.3.10 virtual MamdaFieldState Wombat::MamdaOrderBookConcreteComplexDelta::getSymbolFieldState () const
    [virtual]

Get the string symbol field state.

Returns:
    MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.81.3.11 virtual MamdaFieldState Wombat::MamdaOrderBookConcreteComplexDelta::getPartIdFieldState () const
    [virtual]

Get the participant identifier field state.

Returns:
    MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.81.3.12 virtual MamdaFieldState Wombat::MamdaOrderBookConcreteComplexDelta::getEventSeqNumFieldState () const
    [virtual]

Get the event sequence number field state.

Returns:
    MamdaFieldState. An enumeration representing field state.
Implements `Wombat::MamdaBasicEvent`.

### 6.81.3.13 virtual `MamdaFieldState` Wombat::MamdaOrderBookConcreteComplexDelta::getEventTimeFieldState () const

Get the event time field state.

**Returns:**

`MamdaFieldState`. An enumeration representing field state.

Implements `Wombat::MamdaBasicEvent`.

### 6.81.3.14 virtual `MamdaFieldState` Wombat::MamdaOrderBookConcreteComplexDelta::getSrcTimeFieldState () const

Get the source time field state.

**Returns:**

`MamdaFieldState`. An enumeration representing field state.

Implements `Wombat::MamdaBasicEvent`.

### 6.81.3.15 virtual `MamdaFieldState` Wombat::MamdaOrderBookConcreteComplexDelta::getActivityTimeFieldState () const

Get the activity time field state.

**Returns:**

`MamdaFieldState`. An enumeration representing field state.

Implements `Wombat::MamdaBasicEvent`.

### 6.81.3.16 virtual `MamdaFieldState` Wombat::MamdaOrderBookConcreteComplexDelta::getLineTimeFieldState () const

Get the line time of the update.
6.81 Wombat::MamdaOrderBookConcreteComplexDelta Class Reference

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.81.3.17 virtual MamdaFieldState Wombat::MamdaOrderBookConcreteComplexDelta::getSendTimeFieldState () const

Get the send time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.81.3.18 virtual MamdaFieldState Wombat::MamdaOrderBookConcreteComplexDelta::getMsgQualFieldState () const

Get the message qualifier field state.

Returns:

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.
6.81.3.19 virtual void Wombat::MamdaOrderBookConcreteComplexDelta::setSymbol (const char * value) [virtual]

6.81.3.20 virtual void Wombat::MamdaOrderBookConcreteComplexDelta::setPartId (const char * value) [virtual]

6.81.3.21 virtual void Wombat::MamdaOrderBookConcreteComplexDelta::setEventSeqNum (mama_seqnum_t value) [virtual]

6.81.3.22 virtual void Wombat::MamdaOrderBookConcreteComplexDelta::setEventTime (const MamaDateTime & value) [virtual]

6.81.3.23 virtual void Wombat::MamdaOrderBookConcreteComplexDelta::setSrcTime (const MamaDateTime & value) [virtual]

6.81.3.24 virtual void Wombat::MamdaOrderBookConcreteComplexDelta::setActivityTime (const MamaDateTime & value) [virtual]

6.81.3.25 virtual void Wombat::MamdaOrderBookConcreteComplexDelta::setLineTime (const MamaDateTime & value) [virtual]

6.81.3.26 virtual void Wombat::MamdaOrderBookConcreteComplexDelta::setSendTime (const MamaDateTime & value) [virtual]

6.81.3.27 virtual void Wombat::MamdaOrderBookConcreteComplexDelta::setMsgQual (const MamaMsgQual & value) [virtual]

The documentation for this class was generated from the following file:

- MamdaOrderBookConcreteComplexDelta.h
6.82 Wombat::MamdaOrderBookConcreteSimpleDelta Class Reference

MamdaOrderBookConcreteSimpleDelta is a class that saves information about a simple order book delta.

#include <MamdaOrderBookConcreteSimpleDelta.h>

Inheritance diagram for Wombat::MamdaOrderBookConcreteSimpleDelta::

```
Wombat::MamdaOrderBookBasicDelta
  ↓
Wombat::MamdaOrderBookSimpleDelta
  ↓
Wombat::MamdaOrderBookConcreteSimpleDelta
```

Public Member Functions

- MamdaOrderBookConcreteSimpleDelta ()
- virtual ~MamdaOrderBookConcreteSimpleDelta ()
- virtual const char * getSymbol () const
  
  Get the instruments string symbol.

- virtual const char * getPartId () const
  
  Get the participant identifier.

- virtual mama_seqnum_t getEventSeqNum () const
  
  Get the event sequence number.

- virtual const MamaDateTime & getEventTime () const
  
  Get the event time.

- virtual const MamaDateTime & getSrcTime () const
  
  Get the source time.

- virtual const MamaDateTime & getActivityTime () const
  
  Get the activity time.

- virtual const MamaDateTime & getLineTime () const
  
  Get the line time.
• virtual const MamaDateTime & getSendTime () const
  
  Get the send time.

• virtual const MamaMsgQual & getMsgQual () const
  
  Get the message qualifier.

• virtual MamdaFieldState getSymbolFieldState () const
  
  Get the string symbol field state.

• virtual MamdaFieldState getPartIdFieldState () const
  
  Get the participant identifier field state.

• virtual MamdaFieldState getEventSeqNumFieldState () const
  
  Get the event sequence number field state.

• virtual MamdaFieldState getEventTimeFieldState () const
  
  Get the event time field state.

• virtual MamdaFieldState getSrcTimeFieldState () const
  
  Get the source time field state.

• virtual MamdaFieldState getActivityTimeFieldState () const
  
  Get the activity time field state.

• virtual MamdaFieldState getLineTimeFieldState () const
  
  Get the line time of the update.

• virtual MamdaFieldState getSendTimeFieldState () const
  
  Get the send time field state.

• virtual MamdaFieldState getMsgQualFieldState () const
  
  Get the message qualifier field state.

• virtual void setSymbol (const char ∗value)
• virtual void setPartId (const char ∗value)
• virtual void setEventSeqNum (mama_seqnum_t value)
• virtual void setEventTime (const MamaDateTime &value)
• virtual void setSrcTime (const MamaDateTime &value)
• virtual void setActivityTime (const MamaDateTime &value)
• virtual void setLineTime (const MamaDateTime &value)
• virtual void setSendTime (const MamaDateTime &value)
• virtual void setMsgQual (const MamaMsgQual &value)
6.82.1 Detailed Description

*MamdaOrderBookConcreteSimpleDelta* is a class that saves information about a simple order book delta.

A simple delta is one that affects a single order book entry.

6.82.2 Constructor & Destructor Documentation

6.82.2.1 Wombat::MamdaOrderBookConcreteSimpleDelta::MamdaOrderBookConcreteSimpleDelta ()

6.82.2.2 virtual Wombat::MamdaOrderBookConcreteSimpleDelta::~MamdaOrderBookConcreteSimpleDelta () [virtual]

6.82.3 Member Function Documentation

6.82.3.1 virtual const char * Wombat::MamdaOrderBookConcreteSimpleDelta::getSymbol () const [virtual]

Get the instruments string symbol.

**Returns:**

Symbol. This is the "well-known" symbol for the security, including any symbology mapping performed by the publisher.

Implements *Wombat::MamdaBasicEvent*.

6.82.3.2 virtual const char * Wombat::MamdaOrderBookConcreteSimpleDelta::getPartId () const [virtual]

Get the participant identifier.

**Returns:**

Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).

Implements *Wombat::MamdaBasicEvent*. 

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.82.3.3 virtual mama_seqnum_t Wombat::MamdaOrderBookConcreteSimpleDelta::getEventSeqNum () const [virtual]

Get the event sequence number.

Returns:

Source sequence number. The exchange generated sequence number.

Implements Wombat::MamdaBasicEvent.

6.82.3.4 virtual const MamaDateTime& Wombat::MamdaOrderBookConcreteSimpleDelta::getEventTime () const [virtual]

Get the event time.

Returns:

Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.82.3.5 virtual const MamaDateTime& Wombat::MamdaOrderBookConcreteSimpleDelta::getSrcTime () const [virtual]

Get the source time.

Returns:

Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.82.3.6 virtual const MamaDateTime& Wombat::MamdaOrderBookConcreteSimpleDelta::getActivityTime () const [virtual]

Get the activity time.
6.82 Wombat::MamdaOrderBookConcreteSimpleDelta Class Reference

Returns:

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements Wombat::MamdaBasicEvent.

6.82.3.7 virtual const MamaDateTime& Wombat::MamdaOrderBookConcreteSimpleDelta::getLineTime () const
[virtual]

Get the line time.

Returns:

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

6.82.3.8 virtual const MamaDateTime& Wombat::MamdaOrderBookConcreteSimpleDelta::getSendTime () const
[virtual]

Get the send time.

Returns:

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime()).

Implements Wombat::MamdaBasicEvent.

6.82.3.9 virtual const MamaMsgQual& Wombat::MamdaOrderBookConcreteSimpleDelta::getMsgQual () const
[virtual]

Get the message qualifier.

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
Returns:
Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.82.3.10 virtual MamdaFieldState Wombat::MamdaOrderBookConcreteSimpleDelta::getSymbolFieldState () const
[virtual]

Get the string symbol field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.82.3.11 virtual MamdaFieldState Wombat::MamdaOrderBookConcreteSimpleDelta::getPartIdFieldState () const
[virtual]

Get the participant identifier field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.82.3.12 virtual MamdaFieldState Wombat::MamdaOrderBookConcreteSimpleDelta::getEventSeqNumFieldState () const
[virtual]

Get the event sequence number field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
virtual MamdaFieldState Wombat::MamdaOrderBookConcreteSimpleDelta::getEventTimeFieldState () const

Get the event time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

virtual MamdaFieldState Wombat::MamdaOrderBookConcreteSimpleDelta::getSrcTimeFieldState () const

Get the source time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

virtual MamdaFieldState Wombat::MamdaOrderBookConcreteSimpleDelta::getActivityTimeFieldState () const

Get the activity time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

virtual MamdaFieldState Wombat::MamdaOrderBookConcreteSimpleDelta::getLineTimeFieldState () const

Get the line time of the update.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.82.3.17 virtual MamdaFieldState Wombat::MamdaOrderBookConcreteSimpleDelta::getSendTimeFieldState () const [virtual]

Get the send time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.82.3.18 virtual MamdaFieldState Wombat::MamdaOrderBookConcreteSimpleDelta::getMsgQualFieldState () const [virtual]

Get the message qualifier field state.

Returns:

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.
6.82 Wombat::MamdaOrderBookConcreteSimpleDelta Class Reference

6.82.3.19 virtual void Wombat::MamdaOrderBookConcreteSimpleDelta::setSymbol (const char * value)
[virtual]

6.82.3.20 virtual void Wombat::MamdaOrderBookConcreteSimpleDelta::setPartId (const char * value)
[virtual]

6.82.3.21 virtual void Wombat::MamdaOrderBookConcreteSimpleDelta::setEventSeqNum (mama_seqnum_t value)
[virtual]

6.82.3.22 virtual void Wombat::MamdaOrderBookConcreteSimpleDelta::setEventTime (const MamaDateTime & value)
[virtual]

6.82.3.23 virtual void Wombat::MamdaOrderBookConcreteSimpleDelta::setSrcTime (const MamaDateTime & value)
[virtual]

6.82.3.24 virtual void Wombat::MamdaOrderBookConcreteSimpleDelta::setActivityTime (const MamaDateTime & value)
[virtual]

6.82.3.25 virtual void Wombat::MamdaOrderBookConcreteSimpleDelta::setLineTime (const MamaDateTime & value)
[virtual]

6.82.3.26 virtual void Wombat::MamdaOrderBookConcreteSimpleDelta::setSendTime (const MamaDateTime & value)
[virtual]

6.82.3.27 virtual void Wombat::MamdaOrderBookConcreteSimpleDelta::setMsgQual (const MamaMsgQual & value)
[virtual]

The documentation for this class was generated from the following file:

- MamaOrderBookConcreteSimpleDelta.h
6.83 Wombat::MamdaOrderBookDelta Class Reference

MamdaOrderBookDelta is an interface that provides access to order book related fields.
#include <MamdaOrderBookDelta.h>

Inheritance diagram for Wombat::MamdaOrderBookDelta::

```
Wombat::MamdaBasicEvent
```
```
Wombat::MamdaOrderBookDelta
```

Public Member Functions

- virtual const MamdaOrderBook & getDeltaOrderBook () const =0
  Return the order book delta.
- virtual ~MamdaOrderBookDelta ()

6.83.1 Detailed Description

MamdaOrderBookDelta is an interface that provides access to order book related fields.

6.83.2 Constructor & Destructor Documentation

6.83.2.1 virtual Wombat::MamdaOrderBookDelta::~MamdaOrderBookDelta () [virtual]

47 {};

6.83.3 Member Function Documentation

6.83.3.1 virtual const MamdaOrderBook& Wombat::MamdaOrderBookDelta::getDeltaOrderBook () const [pure virtual]

Return the order book delta.
In the case of recap order book, the delta will be the entire recap.
Returns:

The order book delta.

The documentation for this class was generated from the following file:

- MamdaOrderBookDelta.h
### 6.84 MamdaOrderBookDepthFilter Class Reference

```cpp
#include <MamdaOrderBookDepthFilter.h>
```

Inheritance diagram for MamdaOrderBookDepthFilter:

```
Wombat::MamdaMsgListener

Wombat::MamdaOrderBookListener Wombat::MamdaOrderBookHandler

MamdaOrderBookDepthFilter
```

#### Public Member Functions

- **MamdaOrderBookDepthFilter** (MamdaOrderBookListener &parent, size_t depth)
- virtual ~MamdaOrderBookDepthFilter ()
- virtual void addHandler (MamdaOrderBookHandler *handler)

  Add a specialized order book handler.

- virtual void removeHandlers ()

  Remove the reference of handlers from the internal list.

- virtual const MamdaOrderBook * getOrderBook () const

  Return the filtered orderbook.

- virtual const MamdaOrderBook * getFullOrderBook () const

  Return the full depth orderbook.

- virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)

  Implementation of MamdaListener interface.

- virtual void onBookRecap (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookComplexDelta *delta, const MamdaOrderBookRecap &event, const MamdaOrderBook &book)

  Method invoked when a full refresh of the order book for the security is available.
6.84 MamdaOrderBookDepthFilter Class Reference

- virtual void onBookDelta (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookSimpleDelta &event, const MamdaOrderBook &book)
  
  Method invoked when a basic order book delta is reported.

- virtual void onBookComplexDelta (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookComplexDelta &event, const MamdaOrderBook &book)
  
  Method invoked when an order book delta is reported.

- virtual void onBookClear (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookClear &event, const MamdaOrderBook &book)
  
  Method invoked when an order book is cleared.

- virtual void onBookGap (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookGap &event, const MamdaOrderBook &book)
  
  Method invoked when a gap in order book updates is discovered.

6.84.1 Constructor & Destructor Documentation

6.84.1.1 MamdaOrderBookDepthFilter::MamdaOrderBookDepthFilter (MamdaOrderBookListener & parent, size_t depth)

6.84.1.2 virtual MamdaOrderBookDepthFilter::~MamdaOrderBookDepthFilter () [virtual]

6.84.2 Member Function Documentation

6.84.2.1 virtual void MamdaOrderBookDepthFilter::addHandler (MamdaOrderBookHandler * handler) [virtual]

Add a specialized order book handler.
Currently, only one handler can (and must) be registered.

Parameters:

handler The hadler registered to receive order book update callbacks.

Reimplemented from Wombat::MamdaOrderBookListener.
6.84.2.2  virtual void MamdaOrderBookDepthFilter::removeHandlers ()
    [virtual]

Remove the reference of handlers from the internal list.
Memory is not freed.
Reimplemented from Wombat::MamdaOrderBookListener.

6.84.2.3  virtual const MamdaOrderBook∗ MamdaOrderBookDepthFilter::getOrderBook () const  [virtual]

Return the filtered orderbook.
Reimplemented from Wombat::MamdaOrderBookListener.

6.84.2.4  virtual const MamdaOrderBook∗ MamdaOrderBookDepthFilter::getFullOrderBook () const
    [virtual]

Return the full depth orderbook.

6.84.2.5  virtual void MamdaOrderBookDepthFilter::onMsg
    (MamdaSubscription ∗ subscription, const MamaMsg & msg, short
    msgType)  [virtual]

Implementation of MamdaListener interface.
Reimplemented from Wombat::MamdaOrderBookListener.

70  { }; 

6.84.2.6  virtual void MamdaOrderBookDepthFilter::onBookRecap (MamdaSubscription ∗ subscription, MamdaOrderBookListener & listener,
    const MamaMsg ∗ msg, const MamdaOrderBookComplexDelta ∗ delta,
    const MamdaOrderBookRecap & event, const MamdaOrderBook &
    book)  [virtual]

Method invoked when a full refresh of the order book for the security is available.
The reason for the invocation may be any of the following:

• Initial image.
• Start-of-day book state.
• Recap update (e.g., after server fault tolerant event or data quality event.)
• After stale status removed.

Parameters:

subscription  The subscription which received the update.
listener  The order book listener that invoked this callback.
msg  The MamaMsg that triggered this invocation.
delta  Always NULL. Reserved for future use.
event  The order book recap event.
book  The current full book.

Implements Wombat::MamdaOrderBookHandler.

virtual void MamdaOrderBookDepthFilter::onBookDelta
(MamdaSubscription * subscription, MamdaOrderBookListener &
listener, const MamaMsg * msg, const MamdaOrderBookSimpleDelta &
& event, const MamdaOrderBook & book) [virtual]

Method invoked when a basic order book delta is reported.
A basic delta consists of one price levels (add/update/delete), which contains zero or
one entries (add/update/delete). Some feeds do not provide order book entry informa-
tion.

Parameters:

subscription  The MamdaSubscription handle.
listener  The order book listener that received the update.
msg  The MamaMsg that triggered this invocation.
event  Details of the delta event.
book  The current full book (after applying the delta).

Implements Wombat::MamdaOrderBookHandler.

virtual void MamdaOrderBookDepthFilter::onBookComplexDelta
(MamdaSubscription * subscription, MamdaOrderBookListener &
listener, const MamaMsg * msg, const MamdaOrderBookComplexDelta &
& event, const MamdaOrderBook & book) [virtual]

Method invoked when an order book delta is reported.
A delta consists of one or more price levels (add/update/delete), each of which contains zero or more entries (add/update/delete). Some feeds do not provide order book entry information.

**Parameters:**

- **subscription** The MamdaSubscription handle.
- **listener** The order book listener that received the update.
- **msg** The MamaMsg that triggered this invocation.
- **event** Details of the the delta.
- **book** The current full book (after applying the delta).

Implements Wombat::MamdaOrderBookHandler.

6.84.2.9 virtual void MamdaOrderBookDepthFilter::onBookClear(MamdaSubscription * subscription, MamdaOrderBookListener & listener, const MamaMsg * msg, const MamdaOrderBookClear & event, const MamdaOrderBook & book) [virtual]

Method invoked when an order book is cleared.

**Parameters:**

- **subscription** The MamdaSubscription handle.
- **listener** The order book listener that caused the clear.
- **msg** The MamaMsg that triggered this invocation.
- **event** The clear event.
- **book** The current state of the book (empty).

Implements Wombat::MamdaOrderBookHandler.

6.84.2.10 virtual void MamdaOrderBookDepthFilter::onBookGap(MamdaSubscription * subscription, MamdaOrderBookListener & listener, const MamaMsg * msg, const MamdaOrderBookGap & event, const MamdaOrderBook & book) [virtual]

Method invoked when a gap in order book updates is discovered.

It is usual for a recap to follow shortly after an order book gap is detected.

**Parameters:**

- **subscription** The MamdaSubscription handle.
6.84 MamdaOrderBookDepthFilter Class Reference

listener The order book listener that detected the gap.
msg The MamaMsg that triggered this invocation.
event Provides the sequence number gap range.
book The current state of the book.

Implements Wombat::MamdaOrderBookHandler.

The documentation for this class was generated from the following file:

- MamdaOrderBookDepthFilter.h
6.85 Wombat::MamdaOrderBookDuplicateEntry
Class Reference

MamdaOrderBookDuplicateEntry is generated when an existing entry is unexpectedly encountered when updating a MamdaOrderBook or MamdaOrderBookEntryManager.

#include <MamdaOrderBookExceptions.h>

Inheritance diagram for Wombat::MamdaOrderBookDuplicateEntry:

```
Wombat::MamdaOrderBookDuplicateEntry
```

Public Member Functions

- **MamdaOrderBookDuplicateEntry** (MamdaOrderBookEntry *origEntry, MamdaOrderBookEntry *dupEntry)
- MamdaOrderBookEntry * getOrigEntry ()
- MamdaOrderBookEntry * getDupEntry ()

6.85.1 Detailed Description

MamdaOrderBookDuplicateEntry is generated when an existing entry is unexpectedly encountered when updating a MamdaOrderBook or MamdaOrderBookEntryManager.

6.85.2 Constructor & Destructor Documentation

6.85.2.1 Wombat::MamdaOrderBookDuplicateEntry::MamdaOrderBookDuplicateEntry (MamdaOrderBookEntry * origEntry, MamdaOrderBookEntry * dupEntry)

```cpp
65 : MamdaOrderBookException (*"duplicate entry ID")
66 , mOrigEntry (origEntry)
67 , mDupEntry (dupEntry)
68 {}```

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.85.3 Member Function Documentation

6.85.3.1 `MamdaOrderBookEntry* Wombat::MamdaOrderBookDuplicateEntry::getOrigEntry()`

69 { return mOrigEntry; }

6.85.3.2 `MamdaOrderBookEntry* Wombat::MamdaOrderBookDuplicateEntry::getDupEntry()`

70 { return mDupEntry; }

The documentation for this class was generated from the following file:

- `MamdaOrderBookExceptions.h`
MamdaOrderBookEntry is a class that represents an entry within a price level of an order book.

```cpp
#include <MamdaOrderBookEntry.h>
```

### Public Types

- `MAMDA_BOOK_ACTION_ADD` = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_ADD
- `MAMDA_BOOK_ACTION_UPDATE` = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_UPDATE
- `MAMDA_BOOK_ACTION_DELETE` = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_DELETE
- `MAMDA_BOOK_ACTION_UNKNOWN` = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_UNKNOWN
- `enum Action { MAMDA_BOOK_ACTION_ADD = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_ADD, MAMDA_BOOK_ACTION_UPDATE = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_UPDATE, MAMDA_BOOK_ACTION_DELETE = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_DELETE, MAMDA_BOOK_ACTION_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_UNKNOWN }`

An enumeration for book entry actions.

### Public Member Functions

- `MamdaOrderBookEntry ()`
- `MamdaOrderBookEntry (const MamdaOrderBookEntry &copy)`
  
  Copy constructor.

- `MamdaOrderBookEntry (const char *entryId, mama_quantity_t size, Action action, const MamaDateTime &entryTime, const MamaSourceDerivative *deriv)`

  Constructor that takes the basic information for the entry ID, size, action, and time.

- `~MamdaOrderBookEntry ()`
- `MamdaOrderBookEntry & operator= (const MamdaOrderBookEntry &rhs)`

  Assignment operator.

- `void clear ()`
- `void copy (const MamdaOrderBookEntry &copy)`
Copy an order book entry.

- void setId (const char∗ id)
- void setUniqueId (const char∗ id)
- void setSize (mama_quantity_t size)
- void setAction (Action action)
- void setReason (MamdaOrderBookTypes::Reason reason)
- void setTime (const MamaDateTime &time)
- void setStatus (mama_u16_t status)
- void setDetails (const MamdaOrderBookEntry &copy)
- const char∗ getId () const
  If supported, Order book entry ID (order ID, participant ID, etc.
- const char∗ getUniqueId () const
  If supported, Order book entry unique ID (order ID, participant ID, etc.

- mama_quantity_t getSize () const
  The size of the order entry.
- Action getAction () const
  Whether to ADD, UPDATE or DELETE the entry.
- MamdaOrderBookTypes::Reason getReason () const
  Reason for a change.
- const MamaDateTime & getTime () const
  Time of order book entry update.
- mama_u16_t getStatus () const
  Get the status.
- double getPrice () const
  Get the price for this entry.
- MamaPrice getMamaPrice () const
- MamdaOrderBookPriceLevel::Side getSide () const
  Get the price for this entry.
- mama_u32_t getPosition (mama_u32_t maxPos=0) const
  Get the position in the order book for this entry.
- bool equalId (const char∗ id) const
Whether two participant ids are equal.

- bool operator==(const MamdaOrderBookEntry &rhs) const
  Equality operator.

- bool operator!=(const MamdaOrderBookEntry &rhs) const
  Non-equality operator.

- void setPriceLevel(MamdaOrderBookPriceLevel *level)
  Set the MamdaOrderBookPriceLevel object to which this entry belongs.

- MamdaOrderBookPriceLevel * getPriceLevel() const
  Get the MamdaOrderBookPriceLevel object to which this entry belongs.

- MamdaOrderBook * getOrderBook() const
  Get the order book for this entry, if possible.

- void setManager(MamdaOrderBookEntryManager *manager)
  Set the MamdaOrderBookEntryManager object to which this entry belongs.

- MamdaOrderBookEntryManager * getManager() const
  Get the MamdaOrderBookEntryManager object to which this entry belongs.

- const char * getSymbol() const
  Get the symbol for this entry, if possible.

- void setSourceDerivative(const MamaSourceDerivative *deriv)
  Set the MamaSourceDerivative for this book entry.

- const MamaSourceDerivative * getSourceDerivative() const
  Return the MamaSourceDerivative for this book entry.

- const MamaSource * getSource() const
  Return the MamaSource for this book entry.

- mamaSourceState getSourceState() const
  Get the source state.

- bool getCheckSourceState() const
  Get whether this order book wants to check the source state.

- void setQuality(mamaQuality quality)
Set the entry-level quality factor.

- `mamaQuality getQuality() const`
  
  Get the entry-level quality factor.

- `bool isVisible() const`
  
  Get whether this entry is "visible" in this book.

- `void setClosure(void *closure)`
  
  Set the order book entry closure handle.

- `void * getClosure() const`
  
  Get the order book entry closure handle.

- `void assertEqual(const MamdaOrderBookEntry &rhs) const`
  
  Order book entry equality verification.

### Static Public Member Functions

- `static void setStrictChecking(bool strict)`
  
  Enforce strict checking of order book modifications (at the expense of some performance).

#### 6.86.1 Detailed Description

`MamdaOrderBookEntry` is a class that represents an entry within a price level of an order book.

In addition to being referenced in a `MamdaOrderBookPriceLevel`, an order book entry may also be stored in a `MamdaOrderBookEntryManager`.

#### 6.86.2 Member Enumeration Documentation

##### 6.86.2.1 `enum Wombat::MamdaOrderBookEntry::Action`

An enumeration for book entry actions.

Price level actions differ from entry actions because, for example, a price level message with `ACTION_UPDATE` may consist of entries with `ACTION_ADD`, `ACTION_UPDATE` or `ACTION_DELETE`. 
6.86.3 Constructor & Destructor Documentation

6.86.3.1 Wombat::MamdaOrderBookEntry::MamdaOrderBookEntry ()

6.86.3.2 Wombat::MamdaOrderBookEntry::MamdaOrderBookEntry (const MamdaOrderBookEntry & copy)

Copy constructor.
Note that the associated price level of the original copy is not copied.

6.86.3.3 Wombat::MamdaOrderBookEntry::MamdaOrderBookEntry (const char * entryId, mama_quantity_t size, MamdaOrderBookEntry::Action action, const MamaDateTime & entryTime, const MamaSourceDerivative * deriv)

Constructor that takes the basic information for the entry ID, size, action and time.

6.86.3.4 Wombat::MamdaOrderBookEntry::~MamdaOrderBookEntry ()

6.86.4 Member Function Documentation

6.86.4.1 MamdaOrderBookEntry & Wombat::MamdaOrderBookEntry::operator= (const MamdaOrderBookEntry & rhs)

Assignment operator.
Note that the associated price level of the original copy is not copied.
Copy an order book entry. 
Note that the associated price level of the original copy is not copied.

6.86.4.12 const char∗ Wombat::MamdaOrderBookEntry::getId () const

If supported, Order book entry ID (order ID, participant ID, etc.)

Returns:

   The entry id

6.86.4.13 const char∗ Wombat::MamdaOrderBookEntry::getUniqueId () const

If supported, Order book entry unique ID (order ID, participant ID, etc.
)

   The unique ID should be unique throughout the order book. If no explicit unique ID
   has been set, then it assumed that the basic ID is unique and that is returned.
Returns:
The unique entry id

6.86.4.14 mama_quantity_t Wombat::MamdaOrderBookEntry::getSize ()
const

The size of the order entry.

Returns:
The size of the order entry.

6.86.4.15 Action Wombat::MamdaOrderBookEntry::getAction () const

Whether to ADD, UPDATE or DELETE the entry.

Returns:
The order entry action.

6.86.4.16 MamdaOrderBookTypes::Reason Wombat::MamdaOrderBook-
Entry::getReason () const

Reason for a change.

Returns:
The order entry reason

6.86.4.17 const MamaDateTime& Wombat::MamdaOrderBookEntry::getTime
() const

Time of order book entry update.

Returns:
The time of the order entry update.
6.86.4.18 mama_u16_t Wombat::MamdaOrderBookEntry::getStatus () const
Get the status.

Returns:
the status.

6.86.4.19 double Wombat::MamdaOrderBookEntry::getPrice () const
Get the price for this entry.
This method will throw a MamdaOrderBookInvalidEntry if no MamdaPriceLevel is
associated with it because order book price information is only stored in MamdaOrder-
BookPriceLevel objects.

Returns:
The price for this entry.

6.86.4.20 MamaPrice Wombat::MamdaOrderBookEntry::getMamaPrice () const

6.86.4.21 MamdaOrderBookPriceLevel::Side Wombat::MamdaOrderBook-
Entry::getSide () const
Get the price for this entry.
This method will throw a MamdaOrderBookInvalidEntry if no MamdaPriceLevel is
associated with it because order book price information is only stored in MamdaOrder-
BookPriceLevel objects.

Returns:
The side for this entry.

6.86.4.22 mama_u32_t Wombat::MamdaOrderBookEntry::getPosition
(mama_u32_t maxPos = 0) const
Get the position in the order book for this entry.
If maxPos is not zero, then the method will return a result no greater than maxPos. This
is to prevent searching the entire book when only a limited search is necessary. Note:
the logic used in the positional search is to use the number of entries that `MamdaOrderBookPriceLevel::getNumEntries()` returns for price levels above the entry’s price level. -1 is return if the entry is in the book but not currently "visible" (i.e., it is being omitted because the MAMA source is turned off). A `MamdaOrderBookInvalidEntry` is thrown if the entry is not found in the book.

**Parameters:**

`maxPos` The maximum position to return;

**Returns:**

The position of this entry in the order book.

6.86.4.23 bool Wombat::MamdaOrderBookEntry::equalId (const char * id) const

Whether two participant ids are equal.

**Returns:**

true if the two id’s are equal.

6.86.4.24 bool Wombat::MamdaOrderBookEntry::operator== (const MamdaOrderBookEntry & rhs) const

Equality operator.

Two order book entries are equal if their members are identical.

**Parameters:**

`rhs` The order book entry to compare this entry to.

**Returns:**

Whether the two entries are equal.

6.86.4.25 bool Wombat::MamdaOrderBookEntry::operator!= (const MamdaOrderBookEntry & rhs) const

Non-equality operator.

Two order book entries are equal if their members are identical.
Parameters:

\textit{rhs} The order book entry to compare this entry to.

Returns:

Whether the two entries are not equal.

\begin{verbatim}
226   { return ! operator== (rhs); }
\end{verbatim}

6.86.4.26 \textbf{void Wombat::MamdaOrderBookEntry::setPriceLevel (MamdaOrderBookPriceLevel *} level*)

Set the \textit{MamdaOrderBookPriceLevel} object to which this entry belongs.

This method is invoked automatically internally, by the MAMDA API, when an entry is added to a price level.

Parameters:

\textit{level} The price level to be associated with.

6.86.4.27 \textbf{MamdaOrderBookPriceLevel* Wombat::MamdaOrderBookEntry::getPriceLevel () const}

Get the \textit{MamdaOrderBookPriceLevel} object to which this entry belongs.

Returns:

The price level currently associated with this entry.

6.86.4.28 \textbf{MamdaOrderBook* Wombat::MamdaOrderBookEntry::getOrderBook () const}

Get the order book for this entry, if possible.

This can only be done if the entry is part of a price level and the price level is part of an order book. NULL is returned if no order book can be found.

Returns:

The order book or NULL.
void Wombat::MamdaOrderBookEntry::setManager(MamdaOrderBookEntryManager *manager)

Set the MamdaOrderBookEntryManager object to which this entry belongs.
This method is invoked automatically internally, by the MAMDA API, when an entry
is added to an entry manager.

Parameters:

manager pointer to the MamdaOrderBookEntryManager

MamdaOrderBookEntryManager* Wombat::MamdaOrderBookEntry::getManager() const

Get the MamdaOrderBookEntryManager object to which this entry belongs.

Returns:

The manager currently associated with this entry.

const char* Wombat::MamdaOrderBookEntry::getSymbol() const

Get the symbol for this entry, if possible.
This can only be done if the entry is part of a price level and the price level is part of
an order book. NULL is returned if no symbol can be found.

Returns:

The symbol or NULL.

void Wombat::MamdaOrderBookEntry::setSourceDerivative(const MamaSourceDerivative* deriv)

Set the MamaSourceDerivative for this book entry.
The source derivative is used to help determine what the quality of order book entry is
and to efficiently identify all of the entries for a given source (e.g. for aggregated order
books).

Parameters:

deriv The MAMA source derivative
6.86.4.33 const MamaSourceDerivative* Wombat::MamdaOrderBookEntry::getSourceDerivative () const

Return the MamaSourceDerivative for this book entry.

6.86.4.34 const MamaSource* Wombat::MamdaOrderBookEntry::getSource () const

Return the MamaSource for this book entry.

6.86.4.35 mamaSourceState Wombat::MamdaOrderBookEntry::getSourceState () const

Get the source state.

Returns:

The current MAMA source state.

6.86.4.36 bool Wombat::MamdaOrderBookEntry::getCheckSourceState () const

Get whether this order book wants to check the source state.

Returns:

Whether to check source state.

6.86.4.37 void Wombat::MamdaOrderBookEntry::setQuality (mamaQuality quality)

Set the entry-level quality factor.

This level, if not MAMA_QUALITY_OK, overrides the source-level level.

Parameters:

quality The new entry-level MAMA quality level.
6.86.4.38 mamaQuality Wombat::MamdaOrderBookEntry::getQuality () const

Get the entry-level quality factor.

If the entry-level quality is MAMA_QUALITY_OK, then this method returns the source-level quality.

**Returns:**

The current MAMA quality level.

6.86.4.39 bool Wombat::MamdaOrderBookEntry::isVisible () const

Get whether this entry is "visible" in this book.

Visibility is controlled by the status of the MamaSourceDerivative for the entry.

**Returns:**

Whether the entry is visible.

6.86.4.40 void Wombat::MamdaOrderBookEntry::setClosure (void ∗ closure)

Set the order book entry closure handle.

**Parameters:**

- `closure` The closure.

6.86.4.41 void Wombat::MamdaOrderBookEntry::getClosure () const

Get the order book entry closure handle.

**Returns:**

The entry closure.

6.86.4.42 void Wombat::MamdaOrderBookEntry::assertEqual (const MamdaOrderBookEntry & rhs) const

Order book entry equality verification.

A *MamdaOrderBookException* is thrown if the entries within a price level are not equal, along with the reason for the inequality.
Parameters:

- **rhs** The entry which this entry is being compared to.

Exceptions:

- `<MamdaOrderBookException>` Exception thrown if any errors encountered during book processing.

6.86.4.43 static void Wombat::MamdaOrderBookEntry::setStrictChecking (bool strict) [static]

Enforce strict checking of order book modifications (at the expense of some performance).

This setting is automatically updated by `MamdaOrderBook::setStrictChecking()`.

Parameters:

- **strict** Whether strict checking should be employed.

The documentation for this class was generated from the following file:

- `MamdaOrderBookEntry.h`
#include <MamdaOrderBookEntryFilter.h>

Public Member Functions

- virtual bool checkEntry (MamdaOrderBookEntry *entry)=0
- virtual ~MamdaOrderBookEntryFilter ()

6.87.1 Constructor & Destructor Documentation

6.87.1.1 virtual Wombat::MamdaOrderBookEntryFilter::~MamdaOrderBookEntryFilter () [virtual]

39 {
};

6.87.2 Member Function Documentation

6.87.2.1 virtual bool Wombat::MamdaOrderBookEntryFilter::checkEntry (MamdaOrderBookEntry * entry) [pure virtual]

The documentation for this class was generated from the following file:

- MamdaOrderBookEntryFilter.h
6.88 Wombat::MamdaOrderBookEntryManager Class Reference

MamdaOrderBookEntryManager is a class that provides a global order book lookup service, matching entry IDs that are unique across a set of order books.

#include <MamdaOrderBookEntryManager.h>

Public Member Functions

- **MamdaOrderBookEntryManager** (mama_u32_t approxCount)
  Default constructor.

- **~MamdaOrderBookEntryManager** ()
  Destructor.

- void **clear** ()
  Clear all entries from the manager.

- void **clear** (const char *symbol)
  Clear all entries for a particular symbol from the manager.

- void **addEntry** (MamdaOrderBookEntry *entry)
  Add an entry to the manager.

- void **addEntry** (MamdaOrderBookEntry *entry, const char *entryId)
  Add an entry to the manager using a specific entry ID, which may be different from the entry's entry ID.

- **MamdaOrderBookEntry** * **findEntry** (const char *entryId, bool mustExist)
  Find an entry in the manager.

- void **removeEntry** (const char *entryId)
  Remove an entry from the manager.

- void **removeEntry** (MamdaOrderBookEntry *entry)
  Remove an entry from the manager.

- void **dump** (std::ostream &output) const
  Dump (print) all of the entries in this manager to the output stream.
6.88.1 Detailed Description

MamdaOrderBookEntryManager is a class that provides a global order book lookup service, matching entry IDs that are unique across a set of order books.

Some data sources do not provide a symbol when sending updated or deletions for order book entries. The primary purpose of this class is to return access to the actual order book entry represented by a unique entry ID and, therefore, also (indirectly, via methods available from the book entry object) the order book itself.

6.88.2 Constructor & Destructor Documentation

6.88.2.1 Wombat::MamdaOrderBookEntryManager::MamdaOrderBookEntryManager (mama_u32_t approxCount)

Default constructor.
Create an empty order book entry manager.

6.88.2.2 Wombat::MamdaOrderBookEntryManager::~MamdaOrderBookEntryManager ()

Destructor.

6.88.3 Member Function Documentation

6.88.3.1 void Wombat::MamdaOrderBookEntryManager::clear ()

Clear all entries from the manager.

6.88.3.2 void Wombat::MamdaOrderBookEntryManager::clear (const char * symbol)

Clear all entries for a particular symbol from the manager.

6.88.3.3 void Wombat::MamdaOrderBookEntryManager::addEntry (MamdaOrderBookEntry * entry)

Add an entry to the manager.
This method may throw a MamdaOrderBookDuplicateEntry exception.
6.88 Wombat::MamdaOrderBookEntryManager Class Reference

6.88.3.4 void Wombat::MamdaOrderBookEntryManager::addEntry
   (MamdaOrderBookEntry * entry, const char * entryId)

Add an entry to the manager using a specific entry ID, which may be different from the
entry’s entry ID.
This method may throw a MamdaOrderBookDuplicateEntry exception.

6.88.3.5 MamdaOrderBookEntry* Wombat::MamdaOrderBook-
   EntryManager::findEntry (const char * entryId, bool
   mustExist)

Find an entry in the manager.
If no entry matches the unique entry ID and "mustExist" is true then a MamdaOrder-
BookMissingEntry exception is thrown, otherwise it returns NULL.

6.88.3.6 void Wombat::MamdaOrderBookEntryManager::removeEntry (const
   char * entryId)

Remove an entry from the manager.
This method does not actually delete the MamdaOrderBookEntry object itself.

6.88.3.7 void Wombat::MamdaOrderBookEntryManager::removeEntry
   (MamdaOrderBookEntry * entry)

Remove an entry from the manager.
This method does not actually delete the MamdaOrderBookEntry object itself.

6.88.3.8 void Wombat::MamdaOrderBookEntryManager::dump (std::ostream
   & output) const

Dump (print) all of the entries in this manager to the output stream.
The documentation for this class was generated from the following file:

   • MamdaOrderBookEntryManager.h
MamdaOrderBookException is generated when an inconsistent state is detected in an order book.

```
#include <MamdaOrderBookExceptions.h>
```

Inheritance diagram for Wombat::MamdaOrderBookException:

```
Wombat::MamdaOrderBookException
Wombat::MamdaOrderBookDuplicateEntry
Wombat::MamdaOrderBookInvalidEntry
Wombat::MamdaOrderBookMissingEntry
```

Public Member Functions

- `MamdaOrderBookException (const string &msg)`
- virtual `~MamdaOrderBookException () throw ()`

6.89.1 Detailed Description

MamdaOrderBookException is generated when an inconsistent state is detected in an order book.

Such a condition would usually indicate a potential problem with the publisher’s state of the order book or undetected lost messages over the middleware.

6.89.2 Constructor & Destructor Documentation

6.89.2.1 Wombat::MamdaOrderBookException::MamdaOrderBookException (const string & msg)

```
48 : invalid_argument (msg)
49 {}
```

6.89.2.2 virtual Wombat::MamdaOrderBookException::~MamdaOrderBookException () throw () [virtual]

```
50 {}
```

The documentation for this class was generated from the following file:
• MamdaOrderBookExceptions.h
6.90 Wombat::MamdaOrderBookFields Class Reference

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing orderbook related fields from update messages.

#include <MamdaOrderBookFields.h>

**Static Public Member Functions**

- static void setDictionary (const MamaDictionary &dictionary)
- static void reset ()
- static mama_u16_t getMaxFid ()
- static bool isSet ()
- static int getNumLevelFields ()
  
  *Internal.*

- static int getNumEntryFields ()
  
  *Internal.*

- static bool getHasVectorFields ()
  
  *Internal.*

- static bool getHasFixedLevelFields ()
  
  *Internal.*

- static const MamaDictionary & getDictionary ()
  
  *Get the dictionary used to setup the fields.*

**Static Public Attributes**

- static const MamaFieldDescriptor * BOOK_TIME
- static const MamaFieldDescriptor * NUM_LEVELS
- static const MamaFieldDescriptor * BOOK_PROPERTIES
- static const MamaFieldDescriptor * BOOK_TYPE
- static const MamaFieldDescriptor * BOOK_PROP_MSG_TYPE
- static const MamaFieldDescriptor * PRICE_LEVELS
- static const MamaFieldDescriptor * PL_ACTION
- static const MamaFieldDescriptor * PL_PRICE
- static const MamaFieldDescriptor * PL_SIDE
- static const MamaFieldDescriptor * PL_SIZE
• static const MamaFieldDescriptor * PL_SIZE_CHANGE
• static const MamaFieldDescriptor * PL_TIME
• static const MamaFieldDescriptor * PL_NUM_ENTRIES
• static const MamaFieldDescriptor * PL_NUM_ATTACH
• static const MamaFieldDescriptor * PL_ENTRIES
• static const MamaFieldDescriptor * PL_PROPERTIES
• static const MamaFieldDescriptor * PL_PROP_MSG_TYPE
• static const MamaFieldDescriptor * ENTRY_ID
• static const MamaFieldDescriptor * ENTRY_ACTION
• static const MamaFieldDescriptor * ENTRY_REASON
• static const MamaFieldDescriptor * ENTRY_SIZE
• static const MamaFieldDescriptor * ENTRY_TIME
• static const MamaFieldDescriptor * ENTRY_STATUS
• static const MamaFieldDescriptor * ENTRY_PROPERTIES
• static const MamaFieldDescriptor * ENTRY_PROP_MSG_TYPE
• static const MamaFieldDescriptor * BID_MARKET_ORDERS
• static const MamaFieldDescriptor * ASK_MARKET_ORDERS
• static const MamaFieldDescriptor ** PRICE_LEVEL
• static int PRICE_LEVEL_LENGTH
• static const MamaFieldDescriptor ** PL_ENTRY
• static int PL_ENTRY_LENGTH
• static const MamaFieldDescriptor * BOOK_CONTRIBUTORS

6.90.1 Detailed Description

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing orderbook related fields from update messages.

This class should be initialized prior to using the MamdaOrderBookListener by calling setDictionary() with a valid dictionary object which contains orderbook related fields.
6.90.2  Member Function Documentation

6.90.2.1 static void Wombat::MamdaOrderBookFields::setDictionary (const MamaDictionary & dictionary)  [static]

6.90.2.2 static void Wombat::MamdaOrderBookFields::reset ()  [static]

6.90.2.3 static mama_u16_t Wombat::MamdaOrderBookFields::getMaxFid ()  [static]

6.90.2.4 static bool Wombat::MamdaOrderBookFields::isSet ()  [static]

6.90.2.5 static int Wombat::MamdaOrderBookFields::getNumLevelFields ()  [static]

Internal.

Returns the number of fixed (non-vector) price level fields.

6.90.2.6 static int Wombat::MamdaOrderBookFields::getNumEntryFields ()  [static]

Internal.

Returns the number of fixed (non-vector) entry fields.

6.90.2.7 static bool Wombat::MamdaOrderBookFields::getHasVectorFields ()  [static]

Internal.

Returns whether the dictionary contains vector fields (important when unpacking order book messages).

6.90.2.8 static bool Wombat::MamdaOrderBookFields::getHasFixedLevelFields ()  [static]

Internal.

Returns whether the dictionary contains fixed price level fields (important when unpacking order book messages).
Get the dictionary used to setup the fields.
6.90 Wombat::MamdaOrderBookFields Class Reference

6.90.3 Member Data Documentation

6.90.3.1 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::BOOK_TIME [static]

6.90.3.2 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::NUM_LEVELS [static]

6.90.3.3 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::BOOK_PROPERTIES [static]

6.90.3.4 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::BOOK_TYPE [static]

6.90.3.5 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::BOOK_PROP_MSG_TYPE [static]

6.90.3.6 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PRICE_LEVELS [static]

6.90.3.7 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_ACTION [static]

6.90.3.8 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_PRICE [static]

6.90.3.9 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_SIDE [static]

6.90.3.10 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_SIZE [static]

6.90.3.11 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_SIZE_CHANGE [static]

6.90.3.12 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_TIME [static]

6.90.3.13 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_NUM_ENTRIES [static]

6.90.3.14 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_NUM_ATTACH [static]

6.90.3.15 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_ENTRIES [static]

6.90.3.16 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_PROPERTIES [static]

6.90.3.17 const MamaFieldDescriptor* Wombat::MamdaOrderBookFields::PL_PROP_MSG_TYPE [static]
• MamdaOrderBookFields.h
MamdaOrderBookGap is an interface that provides access to order book gap related fields.

#include <MamdaOrderBookGap.h>

Inheritance diagram for Wombat::MamdaOrderBookGap:

```
Wombat::MamdaOrderBookGap
```

Public Member Functions

- virtual mama_seqnum_t `getBeginGapSeqNum ()` const =0
  
  Beginning sequence number in a detected gap event.

- virtual mama_seqnum_t `getEndGapSeqNum ()` const =0
  
  Ending sequence number in a detected gap event.

- virtual ~MamdaOrderBookGap ()

6.91.1 Detailed Description

MamdaOrderBookGap is an interface that provides access to order book gap related fields.

6.91.2 Constructor & Destructor Documentation

6.91.2.1 virtual Wombat::MamdaOrderBookGap::~MamdaOrderBookGap ()

53 {};

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.91.3 Member Function Documentation

6.91.3.1 virtual mama_seqnum_t Wombat::MamdaOrderBookGap::getBeginGapSeqNum () const [pure virtual]

Beginning sequence number in a detected gap event.

Returns:

The starting sequence number of the gap.

6.91.3.2 virtual mama_seqnum_t Wombat::MamdaOrderBookGap::getEndGapSeqNum () const [pure virtual]

Ending sequence number in a detected gap event.

Returns:

The ending sequence number of the gap.

The documentation for this class was generated from the following file:

- MamdaOrderBookGap.h
MamdaOrderBookHandler is an interface for applications that want to have an easy way to handle order book updates.

#include <MamdaOrderBookHandler.h>

Inheritance diagram for Wombat::MamdaOrderBookHandler:

```
Wombat::MamdaOrderBookHandler
   `--------`
        |   MamdaOrderBookDepthFilter
```

Public Member Functions

- virtual void onBookRecap (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookComplexDelta *delta, const MamdaOrderBookRecap &event, const MamdaOrderBook &book)=0
  
  Method invoked when a full refresh of the order book for the security is available.

- virtual void onBookDelta (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookSimpleDelta &event, const MamdaOrderBook &book)=0
  
  Method invoked when a basic order book delta is reported.

- virtual void onBookComplexDelta (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookComplexDelta &event, const MamdaOrderBook &book)=0
  
  Method invoked when an order book delta is reported.

- virtual void onMarketOrderDelta (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookSimpleDelta &event, const MamdaOrderBook &book)
  
  Deprecated.

- virtual void onMarketOrderComplexDelta (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookComplexDelta &event, const MamdaOrderBook &book)
  
  Deprecated.
• virtual void onBookClear (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookClear &event, const MamdaOrderBook &book)=0
  
  *Method invoked when an order book is cleared.*

• virtual void onBookGap (MamdaSubscription *subscription, MamdaOrderBookListener &listener, const MamaMsg *msg, const MamdaOrderBookGap &event, const MamdaOrderBook &book)=0
  
  *Method invoked when a gap in order book updates is discovered.*

• virtual ~MamdaOrderBookHandler ()

### 6.92.1 Detailed Description

MamdaOrderBookHandler is an interface for applications that want to have an easy way to handle order book updates.

The interface defines callback methods for different types of orderBook-related events: order book recaps and updates.

### 6.92.2 Constructor & Destructor Documentation

#### 6.92.2.1 virtual Wombat::MamdaOrderBookHandler::~MamdaOrderBookHandler () [virtual]

```cpp
172};
```

### 6.92.3 Member Function Documentation

#### 6.92.3.1 virtual void Wombat::MamdaOrderBookHandler::onBookRecap (MamdaSubscription * subscription, MamdaOrderBookListener & listener, const MamaMsg * msg, const MamdaOrderBookComplexDelta * delta, const MamdaOrderBookRecap & event, const MamdaOrderBook & book) [pure virtual]

Method invoked when a full refresh of the order book for the security is available.

The reason for the invocation may be any of the following:

• Initial image.

• Start-of-day book state.
• Recap update (e.g., after server fault tolerant event or data quality event.)
• After stale status removed.

Parameters:
  
  subscription  The subscription which received the update.
  
  listener  The order book listener that invoked this callback.
  
  msg  The MamaMsg that triggered this invocation.
  
  delta  Always NULL. Reserved for future use.
  
  event  The order book recap event.
  
  book  The current full book.

Implemented in MamdaOrderBookDepthFilter.

6.92.3.2  virtual void Wombat::MamdaOrderBookHandler::onBookDelta
  
  (MamdaSubscription *subscription, MamdaOrderBookListener &
   listener, const MamaMsg *msg, const MamdaOrderBookSimpleDelta &
   event, const MamdaOrderBook & book)  [pure virtual]

Method invoked when a basic order book delta is reported.

A basic delta consists of one price levels (add/update/delete), which contains zero or
one entries (add/update/delete). Some feeds do not provide order book entry information.

Parameters:
  
  subscription  The MamdaSubscription handle.
  
  listener  The order book listener that received the update.
  
  msg  The MamaMsg that triggered this invocation.
  
  event  Details of the delta event.
  
  book  The current full book (after applying the delta).

Implemented in MamdaOrderBookDepthFilter.

6.92.3.3  virtual void Wombat::MamdaOrderBookHandler::onBookComplexDelta
  
  (MamdaSubscription *subscription, MamdaOrderBookListener &
   listener, const MamaMsg *msg, const MamdaOrderBookComplexDelta &
   event, const MamdaOrderBook & book)  [pure virtual]

Method invoked when an order book delta is reported.
A delta consists of one or more price levels (add/update/delete), each of which contains zero or more entries (add/update/delete). Some feeds do not provide order book entry information.

**Parameters:**

- `subscription` The `MamdaSubscription` handle.
- `listener` The order book listener that received the update.
- `msg` The `MamaMsg` that triggered this invocation.
- `event` Details of the the delta.
- `book` The current full book (after applying the delta).

Implemented in `MamdaOrderBookDepthFilter`.

**6.92.3.4 virtual void Wombat::MamdaOrderBookHandler::onMarketOrderDelta (MamdaSubscription * subscription, MamdaOrderBookListener & listener, const MamaMsg * msg, const MamdaOrderBookSimpleDelta & event, const MamdaOrderBook & book) [virtual]**

Deprecated.

Market Order updates now available in the `onBookDelta` callback Obtaining the price level from the Delta and calling `getOrderType()` can be used to determine if it is a LIMIT or MARKET level

```cpp
};
```

**6.92.3.5 virtual void Wombat::MamdaOrderBookHandler::onMarketOrderComplexDelta (MamdaSubscription * subscription, MamdaOrderBookListener & listener, const MamaMsg * msg, const MamdaOrderBookComplexDelta & event, const MamdaOrderBook & book) [virtual]**

Deprecated.

Market Order updates now available in the `onBookDelta` callback Obtaining the price level from the Delta and calling `getOrderType()` can be used to determine if it is a LIMIT or MARKET level

```cpp
};
```
6.92.3.6  virtual void Wombat::MamdaOrderBookHandler::onBookClear
(MamdaSubscription * subscription, MamdaOrderBookListener &
listener, const MamaMsg * msg, const MamdaOrderBookClear &
event, const MamdaOrderBook & book) [pure virtual]

Method invoked when an order book is cleared.

Parameters:

subscription  The MamdaSubscription handle.
listener      The order book listener that caused the clear.
msg           The MamaMsg that triggered this invocation.
event         The clear event.
book          The current state of the book (empty).

Implemented in MamdaOrderBookDepthFilter.

6.92.3.7  virtual void Wombat::MamdaOrderBookHandler::onBookGap
(MamdaSubscription * subscription, MamdaOrderBookListener &
listener, const MamaMsg * msg, const MamdaOrderBookGap & event,
const MamdaOrderBook & book) [pure virtual]

Method invoked when a gap in order book updates is discovered.

It is usual for a recap to follow shortly after an order book gap is detected.

Parameters:

subscription  The MamdaSubscription handle.
listener      The order book listener that detected the gap.
msg           The MamaMsg that triggered this invocation.
event         Provides the sequence number gap range.
book          The current state of the book.

Implemented in MamdaOrderBookDepthFilter.

The documentation for this class was generated from the following file:

* MamdaOrderBookHandler.h
6.93 Wombat::MamdaOrderBookInvalidEntry Class Reference

MamdaOrderBookInvalidEntry is generated when an entry is applied as an update to or deletion from an order book but the internal references to price level and/or order book do not exist.

#include <MamdaOrderBookExceptions.h>

Inheritance diagram for Wombat::MamdaOrderBookInvalidEntry:

```
Wombat::MamdaOrderBookInvalidEntry
```

Public Member Functions

- MamdaOrderBookInvalidEntry (const MamdaOrderBookEntry *entry, const string &msg)
- const MamdaOrderBookEntry *getInvalidEntry ()

6.93.1 Detailed Description

MamdaOrderBookInvalidEntry is generated when an entry is applied as an update to or deletion from an order book but the internal references to price level and/or order book do not exist.

6.93.2 Constructor & Destructor Documentation

6.93.2.1 Wombat::MamdaOrderBookInvalidEntry::MamdaOrderBookInvalidEntry (const MamdaOrderBookEntry *entry, const string &msg)

```
109 : MamdaOrderBookException (msg)
110 , mEntry (entry)
111 ()
```
6.93.3 Member Function Documentation

6.93.3.1 const MamdaOrderBookEntry* Wombat::MamdaOrderBookInvalidEntry::getInvalidEntry ()

112 { return mEntry; }

The documentation for this class was generated from the following file:

- MamdaOrderBookExceptions.h
6.94 Wombat::MamdaOrderBookListener Class Reference

MamdaOrderBookListener is a class that specializes in handling order book updates.

```cpp
#include <MamdaOrderBookListener.h>
```

Inheritance diagram for Wombat::MamdaOrderBookListener:

```
Wombat::MamdaMsgListener
Wombat::MamdaOrderBookListener
MamdaOrderBookDepthFilter MamdaQuoteToBookListener
```

Public Member Functions

- **MamdaOrderBookListener (MamdaOrderBook *fullBook=NULL)**
  
  *Create an order book listener using an optional user-provided object for the full order book.*

- virtual `~MamdaOrderBookListener ()`

- virtual void `addHandler (MamdaOrderBookHandler *handler)`
  
  *Add a specialized order book handler.*

- virtual void `setProcessEntries (bool process)`
  
  *Set whether we are interested in "entry level" information at all.*

- virtual bool `getProcessEntries () const`
  
  *Get whether we are interested in "entry level" information.*

- virtual void `setProcessMarketOrders (bool process)`
  
  *Set whether we are interested in market order information, if available.*

- virtual bool `getProcessMarketOrders () const`
  
  *Get whether we are interested in market order information.*

- virtual void `addIgnoreEntryId (const char *id)`
  
  *Add an entry ID to ignore.*
• virtual void removeIgnoreEntryId (const char *id)
  Remove an entry ID to ignore.

• virtual void clear ()
  Clear all cached data fields.

• virtual void removeHandlers ()
  Remove the reference of handlers from the internal list.

• virtual const char * getSymbol () const
• virtual const char * getPartId () const
• virtual const MamaDateTime & getSrcTime () const
• virtual const MamaDateTime & getActivityTime () const
• virtual const MamaDateTime & getLineTime () const
• virtual const MamaDateTime & getSendTime () const
• virtual const MamaMsgQual & getMsgQual () const
• virtual const MamaDateTime & getEventTime () const
• virtual mama_seqnum_t getEventSeqNum () const
• virtual MamdaFieldState getSymbolFieldState () const
• virtual MamdaFieldState getPartIdFieldState () const
• virtual MamdaFieldState getEventSeqNumFieldState () const
• virtual MamdaFieldState getSrcTimeFieldState () const
• virtual MamdaFieldState getActivityTimeFieldState () const
• virtual MamdaFieldState getLineTimeFieldState () const
• virtual MamdaFieldState getSendTimeFieldState () const
• virtual MamdaFieldState getMsgQualFieldState () const
• virtual const MamaOrderBook * getOrderBook () const
• virtual MamaOrderBook * getOrderBook ()
• virtual mama_seqnum_t getBeginGapSeqNum () const
• virtual mama_seqnum_t getEndGapSeqNum () const
• virtual void setUseEntryManager (bool useManager)
  Set whether to use an "entry manager" for finding entries in a book.

• virtual void setEntryIdsAreUnique (bool uniqueIds)
  Set whether the entry IDs are unique across the entire book.

• virtual void setKeepBasicDeltas (bool keep)
  Set whether to gather individual basic deltas when processing a complex delta.

• virtual void setUpdateInconsistentBook (bool update)
  Whether to handle or ignore updates sent for an inconsistent book.
• virtual void setUpdateStaleBook (bool update)
  Whether to handle or ignore updates sent for a stale order book.

• virtual void setClearStaleBook (bool clear)
  Whether to clear the order book upon detection of a stale book condition.

• virtual void setQuality (MamdaSubscription *sub, mamaQuality quality)
  Change the order book quality and invoke appropriate callbacks.

• virtual void setConflateDeltas (bool conflate)
  Whether to conflate the order book deltas.

• virtual bool getConflateDeltas ()
  Whether or not order book deltas are conflated.

• virtual void setConflationInterval (double interval)
  The conflation interval.

• virtual void forceInvokeDeltaHandlers ()
  Invoke delta handlers immediately if there is a conflated delta pending.

• virtual void clearConflatedDeltas ()
  Clear any pending conflated deltas.

• virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)
  Implementation of MamdaListener interface.

• virtual void acquireReadLock ()
  Acquire a read lock on the full book, so that an alternative running thread can view
  the book without it being changed from within the MamdaBookListener.

• virtual void releaseReadLock ()
  Release the read lock on the full book.

• virtual void acquireWriteLock ()
• virtual void releaseWriteLock ()
• virtual void getBookSnapShot (MamdaOrderBook &result)
  A thread safe function that can be called to populate a snapshot of the full book for
  the given symbol providing there already exists a MamdaBookListener for the symbol
  within the process.

• virtual void setBookPropertyFids (mama_fid_t *fids, mama_size_t numFids)
Set the FIDs for any additional fields to cached in the book properties for the book.

- virtual void setLevelPropertyFids (mama_fid_t *fids, mama_size_t numFids)
  
  Set the FIDs for any additional fields to cached in the price level properties for the book.

- virtual void setEntryPropertyFids (mama_fid_t *fids, mama_size_t numFids)
  
  Set the FIDs for any additional fields to cached in the entry properties for the book.

Public Attributes

- MamdaOrderBookListenerImpl & mImpl

6.94.1 Detailed Description

MamdaOrderBookListener is a class that specializes in handling order book updates. Developers provide their own implementation of the MamdaOrderBookHandler interface and will be delivered notifications for order book recaps and deltas. Notifications for order book deltas include the delta itself as well as the full order book with deltas applied. An obvious application for this MAMDA class is any kind of program trading application that looks at depth of book.

Note: The MamdaOrderBookListener class caches the order book. Among other reasons, caching of these fields makes it possible to provide full-book related callbacks, even when the publisher (e.g., feed handler) is only publishing deltas containing modified fields.

6.94.2 Constructor & Destructor Documentation

6.94.2.1 Wombat::MamdaOrderBookListener::MamdaOrderBookListener

(MamdaOrderBook * fullBook = NULL)

Create an order book listener using an optional user-provided object for the full order book.

If "fullBook" is NULL, an object will be allocated internally. If this listener is destroyed then the full order book object will only be destroyed if it was created by the listener (i.e., if fullBook was passed as NULL in this constructor).

Parameters:

  fullBook The MamdaOrderBook used to maintain the full book.
6.94.2.2 virtual Wombat::MamdaOrderBookListener::~MamdaOrderBookListener () [virtual]

6.94.3 Member Function Documentation

6.94.3.1 virtual void Wombat::MamdaOrderBookListener::addHandler
(MamdaOrderBookHandler * handler) [virtual]

Add a specialized order book handler.
Currently, only one handler can (and must) be registered.

Parameters:

handler The handler registered to receive order book update callbacks.

Reimplemented in MamdaOrderBookDepthFilter, and MamdaQuoteToBookListener.

6.94.3.2 virtual void Wombat::MamdaOrderBookListener::setProcessEntries
(bool process) [virtual]

Set whether we are interested in "entry level" information at all.
Many applications only care about price level information. (Default is to process entry level information.)

Parameters:

process Whether to process entries in books.

Reimplemented in MamdaQuoteToBookListener.

6.94.3.3 virtual bool Wombat::MamdaOrderBookListener::getProcessEntries
() const [virtual]

Get whether we are interested in "entry level" information.

Returns:

true if we are set to process entries

6.94.3.4 virtual void Wombat::MamdaOrderBookListener::setProcessMarketOrders
(bool process) [virtual]

Set whether we are interested in market order information, if available.
Parameters:

\[ \text{process} \] Whether to process market order data.

6.94.3.5 virtual bool Wombat::MamdaOrderBookListener::getProcessMarketOrders () const [virtual]

Get whether we are interested in market order information.

Returns:

true if we are set to process market orders

6.94.3.6 virtual void Wombat::MamdaOrderBookListener::addIgnoreEntryId (const char \* id) [virtual]

Add an entry ID to ignore.
This only makes sense when the entry type is participant ID (as opposed to order ID).
All order book updates for this entry ID will be ignored.

Parameters:

\[ \text{id} \] The id of the participant to ignore when process book updates.

Reimplemented in MamdaQuoteToBookListener.

6.94.3.7 virtual void Wombat::MamdaOrderBookListener::removeIgnoreEntryId (const char \* id) [virtual]

Remove an entry ID to ignore.

Parameters:

\[ \text{id} \] The id of the participant whose update will be subsequently processed as part of the book.

See also:

addIgnoreEntryId(const char*)

Reimplemented in MamdaQuoteToBookListener.
6.94.3.8 virtual void Wombat::MamdaOrderBookListener::clear ()
    [virtual]

Clear all cached data fields.
Reimplemented in MamdaQuoteToBookListener.

6.94.3.9 virtual void Wombat::MamdaOrderBookListener::removeHandlers ()
    [virtual]

Remove the reference of handlers from the internal list.
Memory is not freed.
Reimplemented in MamdaOrderBookDepthFilter, and MamdaQuoteToBookListener.

6.94.3.10 virtual const char* Wombat::MamdaOrderBookListener::getSymbol () const
    [virtual]

Reimplemented in MamdaQuoteToBookListener.

6.94.3.11 virtual const char* Wombat::MamdaOrderBookListener::getPartId () const
    [virtual]

Reimplemented in MamdaQuoteToBookListener.

6.94.3.12 virtual const MamaDateTime& Wombat::MamdaOrderBookListener::getSrcTime () const
    [virtual]

Reimplemented in MamdaQuoteToBookListener.

6.94.3.13 virtual const MamaDateTime& Wombat::MamdaOrderBookListener::getActivityTime () const
    [virtual]

Reimplemented in MamdaQuoteToBookListener.

6.94.3.14 virtual const MamaDateTime& Wombat::MamdaOrderBookListener::getLineTime () const
    [virtual]

Reimplemented in MamdaQuoteToBookListener.
6.94 Wombat::MamdaOrderBookListener Class Reference

6.94.3.15 virtual const MamaDateTime& Wombat::MamdaOrderBookListener::getSendTime () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.16 virtual const MamaMsgQual& Wombat::MamdaOrderBookListener::getMsgQual () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.17 virtual const MamaDateTime& Wombat::MamdaOrderBookListener::getEventTime () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.18 virtual mama_seqnum_t Wombat::MamdaOrderBookListener::getEventSeqNum () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.19 virtual MamdaFieldState Wombat::MamdaOrderBookListener::getSymbolFieldState () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.20 virtual MamdaFieldState Wombat::MamdaOrderBookListener::getPartIdFieldState () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.21 virtual MamdaFieldState Wombat::MamdaOrderBookListener::getEventSeqNumFieldState () const [virtual]
Reimplemented in MamdaQuoteToBookListener.
6.94.3.22 virtual MamdaFieldState Wombat::MamdaOrderBookListener::getEventTimeFieldState () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.23 virtual MamdaFieldState Wombat::MamdaOrderBookListener::getSrcTimeFieldState () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.24 virtual MamdaFieldState Wombat::MamdaOrderBookListener::getActivityTimeFieldState () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.25 virtual MamdaFieldState Wombat::MamdaOrderBookListener::getLineTimeFieldState () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.26 virtual MamdaFieldState Wombat::MamdaOrderBookListener::getSendTimeFieldState () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.27 virtual MamdaFieldState Wombat::MamdaOrderBookListener::getMsgQualFieldState () const [virtual]
Reimplemented in MamdaQuoteToBookListener.

6.94.3.28 virtual const MamdaOrderBook* Wombat::MamdaOrderBookListener::getOrderBook () const [virtual]
Reimplemented in MamdaOrderBookDepthFilter, and MamdaQuoteToBookListener.
6.94 Wombat::MamdaOrderBookListener Class Reference

6.94.3.29 virtual MamdaOrderBook* Wombat::MamdaOrderBookListener::getOrderBook () [virtual]

6.94.3.30 virtual mama_seqnum_t Wombat::MamdaOrderBookListener::getBeginGapSeqNum () const [virtual]

Reimplemented in MamdaQuoteToBookListener.

6.94.3.31 virtual mama_seqnum_t Wombat::MamdaOrderBookListener::getEndGapSeqNum () const [virtual]

Reimplemented in MamdaQuoteToBookListener.

6.94.3.32 virtual void Wombat::MamdaOrderBookListener::setUseEntryManager (bool useManager) [virtual]

Set whether to use an "entry manager" for finding entries in a book.

Reimplemented in MamdaQuoteToBookListener.

6.94.3.33 virtual void Wombat::MamdaOrderBookListener::setEntryIdsAreUnique (bool uniqueIds) [virtual]

Set whether the entry IDs are unique across the entire book.

This is only relevant if using an "entry manager".

Reimplemented in MamdaQuoteToBookListener.

6.94.3.34 virtual void Wombat::MamdaOrderBookListener::setKeepBasicDeltas (bool keep) [virtual]

Set whether to gather individual basic deltas when processing a complex delta.

Many applications don’t need the basic deltas and will iterate over part or all of the full book (with the deltas already applied).

Reimplemented in MamdaQuoteToBookListener.
6.94.3.35 virtual void Wombat::MamdaOrderBookListener::setUpdateInconsistentBook (bool update) [virtual]

Whether to handle or ignore updates sent for an inconsistent book.
A book may be in an inconsistent state if there has been a gap on the sequence of update (delta) messages. Default is to not update (and wait for a recap).

Parameters:

 update Whether to apply updates to an inconsistent book.

Reimplemented in MamdaQuoteToBookListener.

6.94.3.36 virtual void Wombat::MamdaOrderBookListener::setUpdateStaleBook (bool update) [virtual]

Whether to handle or ignore updates sent for a stale order book.
A book is in a stale state if its mamaQuality is not MAMA_QUALITY_OK.

Parameters:

 update Whether to apply updates to a stale book.

Reimplemented in MamdaQuoteToBookListener.

6.94.3.37 virtual void Wombat::MamdaOrderBookListener::setClearStaleBook (bool clear) [virtual]

Whether to clear the order book upon detection of a stale book condition.
If this is set to true, then setUpdateStaleBook() should probably have been set to false.
Reimplemented in MamdaQuoteToBookListener.

6.94.3.38 virtual void Wombat::MamdaOrderBookListener::setQuality (MamdaSubscription * sub, mamaQuality quality) [virtual]

Change the order book quality and invoke appropriate callbacks.
Reimplemented in MamdaQuoteToBookListener.
virtual void Wombat::MamdaOrderBookListener::setConflateDeltas (bool conflate) [virtual]

Whether to conflate the order book deltas.

Parameters:

conflate Whether to conflate the order book deltas

Reimplemented in MamdaQuoteToBookListener.

virtual bool Wombat::MamdaOrderBookListener::getConflateDeltas () [virtual]

Whether or not order book deltas are conflated.

Returns:

true if conflation is set to true

Reimplemented in MamdaQuoteToBookListener.

virtual void Wombat::MamdaOrderBookListener::setConflationInterval (double interval) [virtual]

The conflation interval.

Parameters:

interval The conflation interval in seconds. Default is 0.5 seconds.

Reimplemented in MamdaQuoteToBookListener.

virtual void Wombat::MamdaOrderBookListener::forceInvokeDeltaHandlers () [virtual]

Invoke delta handlers immediately if there is a conflated delta pending.

Reimplemented in MamdaQuoteToBookListener.

virtual void Wombat::MamdaOrderBookListener::clearConflatedDeltas () [virtual]

Clear any pending conflated deltas.

Reimplemented in MamdaQuoteToBookListener.
6.94.3.44 virtual void Wombat::MamdaOrderBookListener::onMsg
    (MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [virtual]

Implementation of MamdaListener interface.
Implements Wombat::MamdaMsgListener.
Reimplemented in MamdaOrderBookDepthFilter, and MamdaQuoteToBookListener.

6.94.3.45 virtual void Wombat::MamdaOrderBookListener::acquireReadLock
    () [virtual]

Acquire a read lock on the full book, so that an alternative running thread can view the book without it being changed from within the MamdaBookListener.
releaseReadLock should be called to release the lock.
Reimplemented in MamdaQuoteToBookListener.

6.94.3.46 virtual void Wombat::MamdaOrderBookListener::releaseReadLock
    () [virtual]

Release the read lock on the full book.
A companion to the acquireReadLock method.
Reimplemented in MamdaQuoteToBookListener.

6.94.3.47 virtual void Wombat::MamdaOrderBookListener::acquireWriteLock
    () [virtual]

Reimplemented in MamdaQuoteToBookListener.

6.94.3.48 virtual void Wombat::MamdaOrderBookListener::releaseWriteLock
    () [virtual]

Reimplemented in MamdaQuoteToBookListener.

6.94.3.49 virtual void Wombat::MamdaOrderBookListener::getBookSnapShot
    (MamdaOrderBook & result) [virtual]

A thread safe function that can be called to populate a snapshot of the full book for the given symbol providing there already exists a MamdaBookListener for the symbol within the process.
This routine will throw a `MamdaOrderBookException` in the following three circumstances:

- If the Mamda API hasn't been built with multi-threading enabled, or
- If there is no listener within the process for the given symbol, or
- The listener hasn't as yet received an initial image.

Exceptions:

`<MamdaOrderBookException>` See above for details.

Reimplemented in `MamdaQuoteToBookListener`.

6.94.3.50 virtual void Wombat::MamdaOrderBookListener::setBook-
PropertyFids (mama_fid_t * fids, mama_size_t numFids)  
[virtual]

Set the FIDs for any additional fields to cached in the book properties for the book.

Parameters:

- `fids` the array of FIDs to cache
- `numFids` the size of the array

6.94.3.51 virtual void Wombat::MamdaOrderBookListener::setLevel-
PropertyFids (mama_fid_t * fids, mama_size_t numFids)  
[virtual]

Set the FIDs for any additional fields to cached in the price level properties for the book.

Parameters:

- `fids` the array of FIDs to cache
- `numFids` the size of the array

6.94.3.52 virtual void Wombat::MamdaOrderBookListener::setEntry-
PropertyFids (mama_fid_t * fids, mama_size_t numFids)  
[virtual]

Set the FIDs for any additional fields to cached in the entry properties for the book.
Parameters:

- fids  the array of FIDs to cache
- numFids  the size of the array

6.94.4 Member Data Documentation

6.94.4.1 MamdaOrderBookListenerImpl& Wombat::MamdaOrderBookListener::mImpl

The documentation for this class was generated from the following file:

- MamdaOrderBookListener.h
6.95 Wombat::MamdaOrderBookMissingEntry Class Reference

MamdaOrderBookMissingEntry is generated when an expected entry is not found when updating a MamdaOrderBook or MamdaOrderBookEntryManager.

#include <MamdaOrderBookExceptions.h>

Inheritance diagram for Wombat::MamdaOrderBookMissingEntry::

---

Public Member Functions

- MamdaOrderBookMissingEntry (const string &missingEntryId)
- virtual ~MamdaOrderBookMissingEntry () throw ()
- const char * getMissingEntryId ()

6.95.1 Detailed Description

MamdaOrderBookMissingEntry is generated when an expected entry is not found when updating a MamdaOrderBook or MamdaOrderBookEntryManager.

6.95.2 Constructor & Destructor Documentation

6.95.2.1 Wombat::MamdaOrderBookMissingEntry::MamdaOrderBook-

MissingEntry (const string & missingEntryId)

87 : MamdaOrderBookException ("missing entry ID")
88 , mMissingId {missingEntryId}
89 {}

6.95.2.2 virtual Wombat::MamdaOrderBookMissing-

Entry::~MamdaOrderBookMissingEntry () throw ()
[virtual]

90 {}
6.95.3 Member Function Documentation

6.95.3.1 const char* Wombat::MamdaOrderBookMissingEntry::getMissingEntryId ()

91 { return mMissingId.c_str(); }

The documentation for this class was generated from the following file:

- MamdaOrderBookExceptions.h
MamdaOrderBookPriceLevel is a class that provides a price level type for order books.

```
#include <MamdaOrderBookPriceLevel.h>
```

### Public Types

- `typedef const iterator const_iterator`
- `MAMDA_BOOK_ACTION_ADD = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_ADD`
- `MAMDA_BOOK_ACTION_UPDATE = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_UPDATE`
- `MAMDA_BOOK_ACTION_DELETE = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_DELETE`
- `MAMDA_BOOK_ACTION_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_UNKNOWN`
- `MAMDA_BOOK_SIDE_BID = MamdaOrderBookTypes::MAMDA_BOOK_SIDE_BID`
- `MAMDA_BOOK_SIDE_ASK = MamdaOrderBookTypes::MAMDA_BOOK_SIDE_ASK`
- `MAMDA_BOOK_SIDE_UNKOWN = MamdaOrderBookTypes::MAMDA_BOOK_SIDE_UNKNOWN`
- `MAMDA_BOOK_REASON_MODIFY = MamdaOrderBookTypes::MAMDA_BOOK_REASON_MODIFY`
- `MAMDA_BOOK_REASON_CANCEL = MamdaOrderBookTypes::MAMDA_BOOK_REASON_CANCEL`
- `MAMDA_BOOK_REASON_TRADE = MamdaOrderBookTypes::MAMDA_BOOK_REASON_TRADE`
- `MAMDA_BOOK_REASON_CLOSE = MamdaOrderBookTypes::MAMDA_BOOK_REASON_CLOSE`
- `MAMDA_BOOK_REASON_DROP = MamdaOrderBookTypes::MAMDA_BOOK_REASON_DROP`
- `MAMDA_BOOK_REASON_MISC = MamdaOrderBookTypes::MAMDA_BOOK_REASON_MISC`
- `MAMDA_BOOK_REASON_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_REASON_UNKNOWN`
- `MAMDA_BOOK_LEVEL_LIMIT = MamdaOrderBookTypes::MAMDA_BOOK_LEVEL_LIMIT`
- `MAMDA_BOOK_LEVEL_MARKET = MamdaOrderBookTypes::MAMDA_BOOK_LEVEL_MARKET`
- `MAMDA_BOOK_LEVEL_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_LEVEL_UNKNOWN`
• enum Action {
  MAMDA_BOOK_ACTION_ADD = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_ADD,
  MAMDA_BOOK_ACTION_UPDATE = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_UPDATE,
  MAMDA_BOOK_ACTION_DELETE = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_DELETE,
  MAMDA_BOOK_ACTION_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_UNKNOWN
}

An enumeration for price level actions.

• enum Side {
  MAMDA_BOOK_SIDE_BID = MamdaOrderBookTypes::MAMDA_BOOK_SIDE_BID,
  MAMDA_BOOK_SIDE_ASK = MamdaOrderBookTypes::MAMDA_BOOK_SIDE_ASK,
  MAMDA_BOOK_SIDE_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_SIDE_UNKNOWN
}

An enumeration for the side order book side.

• enum Reason {
  MAMDA_BOOK_REASON_MODIFY = MamdaOrderBookTypes::MAMDA_BOOK_REASON_MODIFY,
  MAMDA_BOOK_REASON_CANCEL = MamdaOrderBookTypes::MAMDA_BOOK_REASON_CANCEL,
  MAMDA_BOOK_REASON_TRADE = MamdaOrderBookTypes::MAMDA_BOOK_REASON_TRADE,
  MAMDA_BOOK_REASON_CLOSE = MamdaOrderBookTypes::MAMDA_BOOK_REASON_CLOSE,
  MAMDA_BOOK_REASON_DROP = MamdaOrderBookTypes::MAMDA_BOOK_REASON_DROP,
  MAMDA_BOOK_REASON_MISC = MamdaOrderBookTypes::MAMDA_BOOK_REASON_MISC,
  MAMDA_BOOK_REASON_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_REASON_UNKNOWN
}

An enumeration for a reason for a change.

• enum OrderType {
  MAMDA_BOOK_LEVEL_LIMIT = MamdaOrderBookTypes::MAMDA_BOOK_LEVEL_LIMIT,
  MAMDA_BOOK_LEVEL_MARKET = MamdaOrderBookTypes::MAMDA_BOOK_LEVEL_MARKET,
  MAMDA_BOOK_LEVEL_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_LEVEL_UNKNOWN
}

An enumeration for the type of level.

Public Member Functions

• MamdaOrderBookPriceLevel ()
• MamdaOrderBookPriceLevel (const MamdaOrderBookPriceLevel &copy)

Copy constructor.
• MamdaOrderBookPriceLevel (double price, Side side)
  Constructor initializing just the price and side.

• MamdaOrderBookPriceLevel (MamaPrice &price, Side side)
• ~MamdaOrderBookPriceLevel ()
• MamdaOrderBookPriceLevel & operator= (const MamdaOrderBookPriceLevel &rhs)
  Assignment operator.

• void copy (const MamdaOrderBookPriceLevel &rhs)
  Copy a price level.

• void copyLevelOnly (const MamdaOrderBookPriceLevel &rhs)
  Copy a price level details only.

• void addEntry (MamdaOrderBookEntry *entry)
  Add an entry to the level (without any sanity checking).

• void updateEntry (const MamdaOrderBookEntry &entry)
  Update an entry by copying the information from another entry.

• void removeEntryById (const MamdaOrderBookEntry &entry)
  Remove an entry based on the entry ID information from another entry.

• void removeEntry (const MamdaOrderBookEntry *entry)
  Remove this precise entry (i.e., based on this object being the exact object in the level).

• void addEntriesFromLevel (const MamdaOrderBookPriceLevel *level, MamdaOrderBookEntryFilter *filter, MamdaOrderBookBasicDeltaList *delta)
  Add all entries from another price level into this level.

• void deleteEntriesFromSource (const MamaSource *source, MamdaOrderBookBasicDeltaList *delta)
  Delete all entries in this price level that have "source" as its MamaSource.

• bool reevaluate ()
  Re-evaluate the price level.

• void clear ()
• void setPrice (double price)
• void setPrice (MamaPrice &price)
• void setSize (mama_quantity_t size)
• void setSizeChange (mama_quantity_t sizeChange)
• void setNumEntries (mama_u32_t numEntries)
• void setSide (Side side)
• void setAction (Action action)
• void setTime (const MamaDateTime &time)
• void setDetails (const MamdaOrderBookPriceLevel &rhs)
• void markAllDeleted ()

   Mark everything in this price level as deleted, including entries.

• void setAsDifference (const MamdaOrderBookPriceLevel &lhs, const MamdaOrderBookPriceLevel &rhs)
• double getPrice () const

   Return the price for this level.

• MamaPrice getMamaPrice () const
• mama_quantity_t getSize () const

   Return the total size (across all entries) for this level.

• mama_quantity_t getSizeChange () const

   Return the size change for this (presumably delta) level.

• mama_u32_t getNumEntries () const

   Return the actual number of entries for this level.

• mama_u32_t getNumEntriesTotal () const

   Return the number of entries that can be iterated over.

• bool empty () const

   Return whether there are no entries for this level.

• Side getSide () const

   Return the side (bid/ask) of the book for this level.

• Action getAction () const

   Return the action for this price level.

• const MamaDateTime & getTime () const

   Return the time stamp for when the price level was last updated.

• bool operator== (const MamdaOrderBookPriceLevel &rhs) const

   Equality operator.

• bool operator!= (const MamdaOrderBookPriceLevel &rhs) const
Non-equality operator.

- **void setOrderBook (MamdaOrderBook *book)**
  
  Set the MamdaOrderBook object to which this price level belongs.

- **MamdaOrderBook * getOrderBook () const**
  
  Get the MamdaOrderBook object to which this entry belongs.

- **const char * getSymbol () const**
  
  Get the symbol for this entry, if possible.

- **MamdaOrderBookEntry * findEntry (const char *id) const**
  
  Return the order book entry with ID "id" in the price level or NULL if not found.

- **MamdaOrderBookEntry * findOrCreateEntry (const char *id)**
  
  Return the order book entry with ID "id" in the price level, creating one if necessary.

- **MamdaOrderBookEntry * findOrCreateEntry (const char *id, bool &newEntry)**
  
  Return the order book entry with ID "id" in the price level, creating one if necessary.

- **MamdaOrderBookEntry * getEntryAtPosition (mama_u32_t pos) const**
  
  Return the order book entry at position "pos" in the price level.

- **void setClosure (void *closure)**
  
  Set the order book price level closure handle.

- **void * getClosure () const**
  
  Get the order book price level closure handle.

- **OrderType getOrderType () const**
  
  Return the order type of the level.

- **void setOrderType (OrderType orderType)**
  
  Set the order type for this level.

- **void assertEqual (const MamdaOrderBookPriceLevel &rhs) const**
  
  Order book price level equality verification.

- **iterator begin ()**
- **const_iterator begin () const**
- **iterator & begin (iterator &reuse)**
- **const_iterator & begin (const_iterator &reuse) const**
• iterator end ()
• const_iterator end () const
• iterator & end (iterator &reuse)
• const_iterator & end (const_iterator &reuse) const
• iterator findEntryAfter (iterator &start, const char *id)
• const_iterator findEntryAfter (const_iterator &start, const char *id) const

Static Public Member Functions

• static void setStrictChecking (bool strict)

Enforce strict checking of order book modifications (at the expense of some performance).

Static Public Attributes

• static bool sortEntriesByTime

Classes

• class iterator

6.96.1 Detailed Description

MamdaOrderBookPriceLevel is a class that provides a price level type for order books.

6.96.2 Member Typedef Documentation

6.96.2.1 typedef const iterator Wombat::MamdaOrderBookPriceLevel::const_iterator

6.96.3 Member Enumeration Documentation

6.96.3.1 enum Wombat::MamdaOrderBookPriceLevel::Action

An enumeration for price level actions.

Price level actions differ from entry actions because, for example, a price level message with ACTION_UPDATE may consist of entries with ACTION_ADD, ACTION_UPDATE or ACTION_DELETE.
### 6.96 Wombat::MamdaOrderBookPriceLevel Class Reference

**Enumerators:**

- `MAMDA_BOOK_ACTION_ADD`
- `MAMDA_BOOK_ACTION_UPDATE`
- `MAMDA_BOOK_ACTION_DELETE`
- `MAMDA_BOOK_ACTION_UNKNOWN`

```cpp
{,
 MAMDA_BOOK_ACTION_ADD = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_ADD,
 MAMDA_BOOK_ACTION_UPDATE = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_UPDATE,
 MAMDA_BOOK_ACTION_DELETE = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_DELETE,
 MAMDA_BOOK_ACTION_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_ACTION_UNKNOWN
};
```

#### 6.96.3.2 enum Wombat::MamdaOrderBookPriceLevel::Side

An enumeration for the side order book side.

"Bid" (or "buy") orders occur on one side and "ask" (or "sell") orders occur on the other.

**Enumerators:**

- `MAMDA_BOOK_SIDE_BID`
- `MAMDA_BOOK_SIDE_ASK`
- `MAMDA_BOOK_SIDE_UNKNOWN`

```cpp
{,
 MAMDA_BOOK_SIDE_BID = MamdaOrderBookTypes::MAMDA_BOOK_SIDE_BID,
 MAMDA_BOOK_SIDE_ASK = MamdaOrderBookTypes::MAMDA_BOOK_SIDE_ASK,
 MAMDA_BOOK_SIDE_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_SIDE_UNKNOWN
};
```

#### 6.96.3.3 enum Wombat::MamdaOrderBookPriceLevel::Reason

An enumeration for a reason for a change.

Some of the values of Reason can mean the same thing, as far as their affect on the order book. If possible, a feed will send MODIFY, CANCEL or TRADE actions so that downstream applications that are interested in such data can handle it; other applications can treat such actions in the same way as an UPDATE action (or as a DELETE action if the size is zero).

**Enumerators:**

- `MAMDA_BOOK_REASON_MODIFY`
MAMDA_BOOK_REASON_CANCEL
MAMDA_BOOK_REASON_TRADE
MAMDA_BOOK_REASON_CLOSE
MAMDA_BOOK_REASON_DROP
MAMDA_BOOK_REASON_MISC
MAMDA_BOOK_REASON_UNKNOWN

83 {
84    MAMDA_BOOK_REASON_MODIFY = MamdaOrderBookTypes::MAMDA_BOOK_REASON_MODIFY,
85    MAMDA_BOOK_REASON_CANCEL = MamdaOrderBookTypes::MAMDA_BOOK_REASON_CANCEL,
86    MAMDA_BOOK_REASON_TRADE = MamdaOrderBookTypes::MAMDA_BOOK_REASON_TRADE,
87    MAMDA_BOOK_REASON_CLOSE = MamdaOrderBookTypes::MAMDA_BOOK_REASON_CLOSE,
88    MAMDA_BOOK_REASON_DROP = MamdaOrderBookTypes::MAMDA_BOOK_REASON_DROP,
89    MAMDA_BOOK_REASON_MISC = MamdaOrderBookTypes::MAMDA_BOOK_REASON_MISC,
90    MAMDA_BOOK_REASON_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_REASON_UNKNOWN
91  };

6.96.3.4 enum Wombat::MamdaOrderBookPriceLevel::OrderType

An enumeration for the type of level.

"Limit" level orders are set at a specific price. "Market" level orders are set at the current market price.

Enumerator:

MAMDA_BOOK_LEVEL_LIMIT
MAMDA_BOOK_LEVEL_MARKET
MAMDA_BOOK_LEVEL_UNKNOWN

98 {
99    MAMDA_BOOK_LEVEL_LIMIT = MamdaOrderBookTypes::MAMDA_BOOK_LEVEL_LIMIT,
100   MAMDA_BOOK_LEVEL_MARKET = MamdaOrderBookTypes::MAMDA_BOOK_LEVEL_MARKET,
101   MAMDA_BOOK_LEVEL_UNKNOWN = MamdaOrderBookTypes::MAMDA_BOOK_LEVEL_UNKNOWN
102  };

6.96.4 Constructor & Destructor Documentation

6.96.4.1 Wombat::MamdaOrderBookPriceLevel::MamdaOrderBookPrice-
Level ()

6.96.4.2 Wombat::MamdaOrderBookPriceLevel::MamdaOrder-
BookPriceLevel (const MamdaOrderBookPriceLevel &
copy)

Copy constructor.
Note that the associated order book of the original copy is not copied.

### 6.96.4.3 Wombat::MamdaOrderBookPriceLevel::MamdaOrderBookPriceLevel (double *price, Side side)

Constructor initializing just the price and side.

### 6.96.4.4 Wombat::MamdaOrderBookPriceLevel::MamdaOrderBookPriceLevel (MamaPrice & price, Side side)

### 6.96.4.5 Wombat::MamdaOrderBookPriceLevel::~MamdaOrderBookPriceLevel ()

### 6.96.5 Member Function Documentation

#### 6.96.5.1 MamdaOrderBookPriceLevel& Wombat::MamdaOrderBookPriceLevel::operator= (const MamdaOrderBookPriceLevel & rhs)

Assignment operator.

Note that the associated order book of the original copy is not copied.

#### 6.96.5.2 void Wombat::MamdaOrderBookPriceLevel::copy (const MamdaOrderBookPriceLevel & rhs)

Copy a price level.

Note that the associated order book of the original copy is not copied.

#### 6.96.5.3 void Wombat::MamdaOrderBookPriceLevel::copyLevelOnly (const MamdaOrderBookPriceLevel & rhs)

Copy a price level details only.

No entries are copied.

#### 6.96.5.4 void Wombat::MamdaOrderBookPriceLevel::addEntry (MamdaOrderBookEntry * entry)

Add an entry to the level (without any sanity checking).
6.96.5.5  void Wombat::MamdaOrderBookPriceLevel::updateEntry (const MamdaOrderBookEntry & entry)

Update an entry by copying the information from another entry.

6.96.5.6  void Wombat::MamdaOrderBookPriceLevel::removeEntryById (const MamdaOrderBookEntry & entry)

Remove an entry based on the entry ID information from another entry.

6.96.5.7  void Wombat::MamdaOrderBookPriceLevel::removeEntry (const MamdaOrderBookEntry * entry)

Remove this precise entry (i.e., based on this object being the exact object in the level).

6.96.5.8  void Wombat::MamdaOrderBookPriceLevel::addEntriesFromLevel (const MamdaOrderBookPriceLevel * level, MamdaOrderBookEntryFilter * filter, MamdaOrderBookBasicDeltaList * delta)

Add all entries from another price level into this level.

Parameters:

level  The price level to add.
filter  Filter to use when adding the entries
delta  An optional delta to collect for the deleted entries.

6.96.5.9  void Wombat::MamdaOrderBookPriceLevel::deleteEntriesFromSource (const MamaSource * source, MamdaOrderBookBasicDeltaList * delta)

Delete all entries in this price level that have "source" as its MamaSource.

Parameters:

source  The source to match.
delta  An optional delta to collect for the deleted entries.
6.96.5.10 bool Wombat::MamdaOrderBookPriceLevel::reevaluate ()

Re-evaluate the price level.
This would be performed after the status of sources and/or subsources of an "aggregated order book" (i.e., a book built from multiple sources) have changed.

Returns:
Whether the book info changed based on the re-evaluation.

6.96.5.11 void Wombat::MamdaOrderBookPriceLevel::clear ()

6.96.5.12 void Wombat::MamdaOrderBookPriceLevel::setPrice (double price)

6.96.5.13 void Wombat::MamdaOrderBookPriceLevel::setPrice (MamaPrice & price)

6.96.5.14 void Wombat::MamdaOrderBookPriceLevel::setSize (mama_quantity_t size)

6.96.5.15 void Wombat::MamdaOrderBookPriceLevel::setSizeChange (mama_quantity_t sizeChange)

6.96.5.16 void Wombat::MamdaOrderBookPriceLevel::setNumEntries (mama_u32_t numEntries)

6.96.5.17 void Wombat::MamdaOrderBookPriceLevel::setSide (Side side)

6.96.5.18 void Wombat::MamdaOrderBookPriceLevel::setAction (Action action)

6.96.5.19 void Wombat::MamdaOrderBookPriceLevel::setTime (const MamaDateTime & time)

6.96.5.20 void Wombat::MamdaOrderBookPriceLevel::setDetails (const MamdaOrderBookPriceLevel & rhs)

6.96.5.21 void Wombat::MamdaOrderBookPriceLevel::markAllDeleted ()

Mark everything in this price level as deleted, including entries.
6.96.5.22 void Wombat::MamdaOrderBookPriceLevel::setAsDifference
(const MamdaOrderBookPriceLevel & lhs, const MamdaOrderBookPriceLevel & rhs)

6.96.5.23 double Wombat::MamdaOrderBookPriceLevel::getPrice () const

Return the price for this level.

Returns:

The price for this level.

6.96.5.24 MamaPrice Wombat::MamdaOrderBookPriceLevel::getMamaPrice () const

6.96.5.25 mama_quantity_t Wombat::MamdaOrderBookPriceLevel::getSize () const

Return the total size (across all entries) for this level.

Returns:

The total size for this level.

6.96.5.26 mama_quantity_t Wombat::MamdaOrderBookPriceLevel::getSizeChange () const

Return the size change for this (presumably delta) level.

This attribute is only of interest for delta order books. For full order books, this field will be equal to the size of the price level.

Returns:

The changed size for this level.

6.96.5.27 mama_u32_t Wombat::MamdaOrderBookPriceLevel::getNumEntries () const

Return the actual number of entries for this level.

The actual number of entries may not equate to the number of entries that can be iterated over if: (a) the feed does not provide the actual entries, or (b) the price level is just a delta.
Returns:

The actual number of entries for this level.

6.96.5.28 mama_u32_t Wombat::MamdaOrderBookPriceLevel::getNumEntriesTotal () const

Return the number of entries that can be iterated over.

Returns:

The number of entries that can be iterated over

6.96.5.29 bool Wombat::MamdaOrderBookPriceLevel::empty () const

Return whether there are no entries for this level.

Returns:

Whether there are no entries for this level.

6.96.5.30 Side Wombat::MamdaOrderBookPriceLevel::getSide () const

Return the side (bid/ask) of the book for this level.

Returns:

The side of the book for this level.

6.96.5.31 Action Wombat::MamdaOrderBookPriceLevel::getAction () const

Return the action for this price level.
All price levels for a full book are marked with ACTION_ADD.

Returns:

The action for this level.
6.96.5.32  const MamaDateTime& Wombat::MamdaOrderBookPriceLevel::getTime () const

Return the time stamp for when the price level was last updated.

**Returns:**

The time stamp for when the price level was last updated.

6.96.5.33  bool Wombat::MamdaOrderBookPriceLevel::operator== (const MamdaOrderBookPriceLevel & rhs) const

Equality operator.
Two price levels are equal if their members and price level entries are identical.

**Parameters:**

`rhs`  The order book price level to compare this level to.

**Returns:**

Whether the price levels are equal.

6.96.5.34  bool Wombat::MamdaOrderBookPriceLevel::operator!= (const MamdaOrderBookPriceLevel & rhs) const

Non-equality operator.
Two price levels are equal if their members and price level entries are identical.

**Parameters:**

`rhs`  The order book price level to compare this level to.

**Returns:**

Whether the price levels are unequal.
6.96.5.35  void Wombat::MamdaOrderBookPriceLevel::setOrderBook
(MamdaOrderBook ∗ book)

Set the MamdaOrderBook object to which this price level belongs.
This method is invoked internally, by the MAMDA API, when a price level is added to a book.

Parameters:

book  The order book to be associated with

6.96.5.36  MamdaOrderBook ∗ Wombat::MamdaOrderBookPriceLevel::getOrderBook () const

Get the MamdaOrderBook object to which this entry belongs.

Returns:

The order book currently associated with this price level.

6.96.5.37  const char ∗ Wombat::MamdaOrderBookPriceLevel::getSymbol () const

Get the symbol for this entry, if possible.
This can only be done if the price level is part of an order book. NULL is returned if no symbol can be found.

Returns:

The symbol or NULL.

6.96.5.38  MamdaOrderBookEntry ∗ Wombat::MamdaOrderBookPriceLevel::findEntry (const char ∗ id) const

Return the order book entry with ID "id" in the price level or NULL if not found.

Parameters:

id  The ID of the order book entry.

Returns:

The order book entry found or NULL if not found.
Return the order book entry with ID "id" in the price level, creating one if necessary.

**Parameters:**

- `id` The ID of the order book entry.

**Returns:**

The order book entry found or created.

Return the order book entry with ID "id" in the price level, creating one if necessary.

**Parameters:**

- `id` The ID of the order book entry.
  - `newEntry` Boolean reference indicating entry creation.

**Returns:**

The order book entry found or created.

Return the order book entry at position "pos" in the price level.

**Parameters:**

- `pos` The position of the order book entry.

**Returns:**

The order book entry or NULL if not found.
6.96.5.42  void Wombat::MamdaOrderBookPriceLevel::setClosure (void *
closure)

Set the order book price level closure handle.

Parameters:

closure  The closure.

6.96.5.43  void* Wombat::MamdaOrderBookPriceLevel::getClosure () const

Get the order book price level closure handle.

Returns:

The closure.

6.96.5.44  OrderType Wombat::MamdaOrderBookPriceLevel::getOrderType ()
const

Return the order type of the level.

Returns:

The order type of the level.

6.96.5.45  void Wombat::MamdaOrderBookPriceLevel::setOrderType
(OrderType orderType)

Set the order type for this level.

Parameters:

orderType  The order type of the level.

6.96.5.46  void Wombat::MamdaOrderBookPriceLevel::assertEqual (const
MamdaOrderBookPriceLevel & rhs) const

Order book price level equality verification.
A MamdaOrderBookException is thrown if the price levels are not equal, along with
the reason for the inequality.
Parameters:

\textit{rhs}  The order book price level to compare this level to.

Exceptions:

\textit{<MamdaOrderBookException>}  If error encountered during book processing.

\textbf{6.96.5.47}  \texttt{iterator} \texttt{Wombat::MamdaOrderBookPriceLevel::begin ()}

\textbf{6.96.5.48}  \texttt{const\_iterator} \texttt{Wombat::MamdaOrderBookPriceLevel::begin () const}

\textbf{6.96.5.49}  \texttt{iterator\&} \texttt{Wombat::MamdaOrderBookPriceLevel::begin (iterator \& reuse)}

\textbf{6.96.5.50}  \texttt{const\_iterator\&} \texttt{Wombat::MamdaOrderBookPriceLevel::begin (const\_iterator \& reuse) const}

\textbf{6.96.5.51}  \texttt{iterator} \texttt{Wombat::MamdaOrderBookPriceLevel::end ()}

\textbf{6.96.5.52}  \texttt{const\_iterator} \texttt{Wombat::MamdaOrderBookPriceLevel::end () const}

\textbf{6.96.5.53}  \texttt{iterator\&} \texttt{Wombat::MamdaOrderBookPriceLevel::end (iterator \& reuse)}

\textbf{6.96.5.54}  \texttt{const\_iterator\&} \texttt{Wombat::MamdaOrderBookPriceLevel::end (const\_iterator \& reuse) const}

\textbf{6.96.5.55}  \texttt{iterator} \texttt{Wombat::MamdaOrderBookPriceLevel::findEntryAfter (iterator \& start, const char \* id)}

\textbf{6.96.5.56}  \texttt{const\_iterator} \texttt{Wombat::MamdaOrderBookPriceLevel::findEntryAfter (const\_iterator \& start, const char \* id) const}

\textbf{6.96.5.57}  \texttt{static void} \texttt{Wombat::MamdaOrderBookPriceLevel::setStrictChecking (bool strict) \ [static]}

Enforce strict checking of order book modifications (at the expense of some performance).

This setting is automatically updated by \texttt{MamdaOrderBook::setStrictChecking()}.  

Parameters:

\textit{strict}  Whether strict checking of modification is enabled.
6.96.6 Member Data Documentation

6.96.6.1 bool Wombat::MamdaOrderBookPriceLevel::sortEntriesByTime
  [static]

The documentation for this class was generated from the following file:

- MamdaOrderBookPriceLevel.h
6.97 Wombat::MamdaOrderBookPriceLevel::iterator Class Reference

#include <MamdaOrderBookPriceLevel.h>

Public Member Functions

- iterator()
- iterator (const iterator &copy)
- iterator (const iteratorImpl &copy)
- ~iterator()
- iterator & operator= (const iterator &rhs)
- iterator & operator++ ()
- const iterator & operator++ () const
- bool operator== (const iterator &rhs) const
- bool operator!= (const iterator &rhs) const
- MamdaOrderBookEntry * operator * ()
- const MamdaOrderBookEntry * operator * () const

Protected Attributes

- iteratorImpl & mImpl

Friends

- class MamdaOrderBookPriceLevel
6.97.1 Constructor & Destructor Documentation

6.97.1.1 Wombat::MamdaOrderBookPriceLevel::iterator::iterator ()

6.97.1.2 Wombat::MamdaOrderBookPriceLevel::iterator::iterator (const iterator & copy)

6.97.1.3 Wombat::MamdaOrderBookPriceLevel::iterator::iterator (const iteratorImpl & copy)

6.97.1.4 Wombat::MamdaOrderBookPriceLevel::iterator::~iterator ()

6.97.2 Member Function Documentation

6.97.2.1 iterator& Wombat::MamdaOrderBookPriceLevel::iterator::operator= (const iterator & rhs)

6.97.2.2 iterator& Wombat::MamdaOrderBookPriceLevel::iterator::operator++ ()

6.97.2.3 const iterator& Wombat::MamdaOrderBookPriceLevel::iterator::operator++ () const

6.97.2.4 bool Wombat::MamdaOrderBookPriceLevel::iterator::operator== (const iterator & rhs) const

6.97.2.5 bool Wombat::MamdaOrderBookPriceLevel::iterator::operator!= (const iterator & rhs) const

6.97.2.6 MamdaOrderBookEntry* Wombat::MamdaOrderBookPriceLevel::iterator::operator* ()

6.97.2.7 const MamdaOrderBookEntry* Wombat::MamdaOrderBookPriceLevel::iterator::operator* () const

6.97.3 Friends And Related Function Documentation

6.97.3.1 friend class MamdaOrderBookPriceLevel [friend]

6.97.4 Member Data Documentation

6.97.4.1 iteratorImpl& Wombat::MamdaOrderBookPriceLevel::iterator::m_Impl [protected]
- MamdaOrderBookPriceLevel.h
6.98 Wombat::MamdaOrderBookRecap Class Reference

MamdaOrderBookRecap is an interface that provides access to order book related fields.

```
#include <MamdaOrderBookRecap.h>
```

Inheritance diagram for Wombat::MamdaOrderBookRecap::

```
Wombat::MamdaBasicEvent
Wombat::MamdaOrderBookRecap
```

Public Member Functions

- virtual const MamdaOrderBook * getOrderBook () const =0
  
  Returns the full orderbook related to this recap event.

- virtual ~MamdaOrderBookRecap ()

6.98.1 Detailed Description

MamdaOrderBookRecap is an interface that provides access to order book related fields.

6.98.2 Constructor & Destructor Documentation

```
6.98.2.1 virtual Wombat::MamdaOrderBookRecap::~MamdaOrderBookRecap () [virtual]
47 {};
```

6.98.3 Member Function Documentation

```
6.98.3.1 virtual const MamdaOrderBook * Wombat::MamdaOrderBookRecap::getOrderBook () const [pure virtual]
```

Returns the full orderbook related to this recap event.
Returns:

The full order book.

The documentation for this class was generated from the following file:

- MamdaOrderBookRecap.h
MamdaOrderBookSimpleDelta is a class that saves information about a simple order book delta.

#include <MamdaOrderBookSimpleDelta.h>

Inheritance diagram for Wombat::MamdaOrderBookSimpleDelta::

![Inheritance Diagram](image)

Public Member Functions

- MamdaOrderBookSimpleDelta (const MamdaOrderBookSimpleDelta &copy)
- MamdaOrderBookSimpleDelta ()
- virtual ~MamdaOrderBookSimpleDelta ()

6.99.1 Detailed Description

MamdaOrderBookSimpleDelta is a class that saves information about a simple order book delta.

A simple delta is one that affects a single order book entry.

6.99.2 Constructor & Destructor Documentation

6.99.2.1 Wombat::MamdaOrderBookSimpleDelta::MamdaOrderBookSimpleDelta (const MamdaOrderBookSimpleDelta &copy)

6.99.2.2 Wombat::MamdaOrderBookSimpleDelta::MamdaOrderBookSimpleDelta ()
6.99.2.3 virtual Wombat::MamdaOrderBookSimpleDelta::~MamdaOrderBookSimpleDelta () [virtual]

45 {}
MamdaOrderBookTypes is a class that provides order book related typed.

#include <MamdaOrderBookTypes.h>

Public Types

- MAMDA_BOOK_ACTION_ADD = 'A'
  A new price level.

- MAMDA_BOOK_ACTION_UPDATE = 'U'
  A updated price level.

- MAMDA_BOOK_ACTION_DELETE = 'D'
  A deleted price level.

- MAMDA_BOOK_SIDE_BID = 'B'
- MAMDA_BOOK_SIDE_ASK = 'A'
  A bid (buy) order.

- MAMDA_BOOK_SIDE_UNKNOWN = 'Z'
  An ask (sell) order.

- MAMDA_BOOK_REASON_MODIFY = 'M'
- MAMDA_BOOK_REASON_CANCEL = 'C'
- MAMDA_BOOK_REASON_TRADE = 'T'
- MAMDA_BOOK_REASON_CLOSE = 'c'
- MAMDA_BOOK_REASON_DROP = 'N'
- MAMDA_BOOK_REASON_MISC = 'm'
- MAMDA_BOOK_REASON_UNKNOWN = 'Z'
- MAMDA_PROP_MSG_TYPE_UPDATE = 'U'
- MAMDA_PROP_MSG_TYPE_RECAP = 'R'
- MAMDA_BOOK_LEVEL_LIMIT = 'L'
- MAMDA_BOOK_LEVEL_MARKET = 'M'
- MAMDA_BOOK_LEVEL_UNKNOWN = 'U'

- enum Action { MAMDA_BOOK_ACTION_ADD = 'A', MAMDA_BOOK_ACTION_UPDATE = 'U', MAMDA_BOOK_ACTION_DELETE = 'D', MAMDA_BOOK_ACTION_UNKNOWN = 'Z' }
  An enumeration for actions on an order book.
enum Side { MAMDA_BOOK_SIDE_BID = 'B', MAMDA_BOOK_SIDE_ASK = 'A', MAMDA_BOOK_SIDE_UNKNOWN = 'Z' }

An enumeration for the side order book side.

enum Reason {
    MAMDA_BOOK_REASON_MODIFY = 'M', MAMDA_BOOK_REASONCANCEL = 'C', MAMDA_BOOK_REASON_TRADE = 'T', MAMDA_BOOK_REASON_CLOSE = 'c',
    MAMDA_BOOK_REASON_DROP = 'N', MAMDA_BOOK_REASON_MISC = 'm', MAMDA_BOOK_REASON_UNKNOWN = 'Z'
}

An enumeration for a reason for a change.

enum PropMsgType { MAMDA_PROP_MSG_TYPE_UPDATE = 'U', MAMDA_PROP_MSG_TYPE_RECAP = 'R' }

An enumeration for property msg types.

enum OrderType { MAMDA_BOOK_LEVEL_LIMIT = 'L', MAMDA_BOOK_LEVEL_MARKET = 'M', MAMDA_BOOK_LEVEL_UNKNOWN = 'U' }

An enumeration for the type of level.

6.100.1 Detailed Description

MamdaOrderBookTypes is a class that provides order book related typed.

6.100.2 Member Enumeration Documentation

6.100.2.1 enum Wombat::MamdaOrderBookTypes::Action

An enumeration for actions on an order book.

Note: price level actions differ from entry actions because, for example, a price level message with ACTION_UPDATE may consist of entries with ACTION_ADD, ACTION_UPDATE or ACTION_DELETE.

Enumerator:

MAMDA_BOOK_ACTION_ADD
MAMDA_BOOK_ACTION_UPDATE A new price level.
MAMDA_BOOK_ACTION_DELETE An updated price level.
MAMDA_BOOK_ACTION_UNKNOWN A deleted price level.
6.100.2.2 enum Wombat::MamdaOrderBookTypes::Side

An enumeration for the side order book side.

"Bid" (or "buy") orders occur on one side and "ask" (or "sell") orders occur on the other.

 Enumerator:

  
  MAMDA_BOOK_SIDE_BID
  MAMDA_BOOK_SIDE_ASK  A bid (buy) order.
  MAMDA_BOOK_SIDE_UNKNOWN  An ask (sell) order.

6.100.2.3 enum Wombat::MamdaOrderBookTypes::Reason

An enumeration for a reason for a change.

Some of the values of Reason can mean the same thing, as far as their affect on the order book. If possible, a feed will send MODIFY, CANCEL or TRADE actions so that downstream applications that are interested in such data can handle it; other applications can treat such actions in the same way as an UPDATE action (or as a DELETE action if the size is zero).

 Enumerator:

  
  MAMDA_BOOK_REASON_MODIFY
  MAMDA_BOOK_REASON_CANCEL
  MAMDA_BOOK_REASON_TRADE
  MAMDA_BOOK_REASON_CLOSE
  MAMDA_BOOK_REASON_DROP
  MAMDA_BOOK_REASON_MISC
  MAMDA_BOOK_REASON_UNKNOWN
enum Wombat::MamdaOrderBookTypes::PropMsgType

An enumeration for property msg types.
They can either be an update or a recap. If the field is not present, update is implied.

Enumerator:

- MAMDA_PROP_MSG_TYPE_UPDATE
- MAMDA_PROP_MSG_TYPE_RECAP

enum Wombat::MamdaOrderBookTypes::OrderType

An enumeration for the type of level.
"Limit" level orders are set at a specific price. "Market" level orders are set at the current market price.

Enumerator:

- MAMDA_BOOK_LEVEL_LIMIT
- MAMDA_BOOK_LEVEL_MARKET
- MAMDA_BOOK_LEVEL_UNKNOWN

The documentation for this class was generated from the following file:

- MamdaOrderBookTypes.h
6.101 Wombat::MamdaOrderImbalanceFields Class Reference

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing imbalance related fields from update messages.

#include <MamdaOrderImbalanceFields.h>

Static Public Member Functions

• static void setDictionary (const MamaDictionary &dictionary)
• static void reset ()

   Reset the dictionary for order imbalance update fields.

• static uint16_t getMaxFid ()
• static bool isSet ()

Static Public Attributes

• static const MamaFieldDescriptor * HIGH_INDICATION_PRICE
• static const MamaFieldDescriptor * LOW_INDICATION_PRICE
• static const MamaFieldDescriptor * INDICATION_PRICE
• static const MamaFieldDescriptor * BUY_VOLUME
• static const MamaFieldDescriptor * SELL_VOLUME
• static const MamaFieldDescriptor * MATCH_VOLUME
• static const MamaFieldDescriptor * SECURITY_STATUS_QUAL
• static const MamaFieldDescriptor * INSIDE_MATCH_PRICE
• static const MamaFieldDescriptor * FAR_CLEARING_PRICE
• static const MamaFieldDescriptor * NEAR_CLEARING_PRICE
• static const MamaFieldDescriptor * NO_CLEARING_PRICE
• static const MamaFieldDescriptor * PRICE_VAR_IND
• static const MamaFieldDescriptor * CROSS_TYPE
• static const MamaFieldDescriptor * SRC_TIME
• static const MamaFieldDescriptor * ACTIVITY_TIME
• static const MamaFieldDescriptor * MSG_TYPE
• static const MamaFieldDescriptor * ISSUE_SYMBOL
• static const MamaFieldDescriptor * PART_ID
• static const MamaFieldDescriptor * SEQ_NUM
• static const MamaFieldDescriptor * SECURITY_STATUS_ORIG
• static const MamaFieldDescriptor * SECURITY_STATUS_TIME
• static const MamaFieldDescriptor * AUCTION_TIME
• static const MamaFieldDescriptor * LINE_TIME
• static const MamaFieldDescriptor * SEND_TIME
6.101 Wombat::MamdaOrderImbalanceFields Class Reference

6.101.1 Detailed Description

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing imbalance related fields from update messages.

This class should be initialized prior to using the MamdaOrderImbalanceListener by calling setDictionary() with a valid dictionary object which contains imbalance related fields.

6.101.2 Member Function Documentation

6.101.2.1 static void Wombat::MamdaOrderImbalanceFields::setDictionary
(const MamaDictionary & dictionary) [static]

6.101.2.2 static void Wombat::MamdaOrderImbalanceFields::reset ()
[static]

Reset the dictionary for order imbalance update fields.
6.101.2.3 static uint16_t Wombat::MamdaOrderImbalanceFields::getMaxFid () [static]

6.101.2.4 static bool Wombat::MamdaOrderImbalanceFields::isSet () [static]

6.101.3 Member Data Documentation

6.101.3.1 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::HIGH_INDICATION_PRICE [static]

6.101.3.2 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::LOW_INDICATION_PRICE [static]

6.101.3.3 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::INDICATION_PRICE [static]

6.101.3.4 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::BUY_VOLUME [static]

6.101.3.5 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::SELL_VOLUME [static]

6.101.3.6 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::MATCH_VOLUME [static]

6.101.3.7 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::SECURITY_STATUS_QUAL [static]

6.101.3.8 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::INSIDE_MATCH_PRICE [static]

6.101.3.9 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::FAR_CLEARING_PRICE [static]

6.101.3.10 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::NEAR_CLEARING_PRICE [static]

6.101.3.11 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::NO_CLEARING_PRICE [static]

6.101.3.12 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::PRICE_VAR_IND [static]

6.101.3.13 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::CROSS_TYPE [static]

6.101.3.14 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::SRC_TIME [static]

6.101.3.15 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::ACTIVITY_TIME [static]

6.101.3.16 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::MSG_TYPE [static]

6.101.3.17 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::ISSUE_SYMBOL [static]

6.101.3.18 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::PART_ID [static]

6.101.3.19 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::SEQ_NUM [static]

6.101.3.20 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::SECURITY_STATUS_ORIG [static]

6.101.3.21 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::SECURITY_STATUS_TIME [static]

6.101.3.22 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::AUCTION_TIME [static]

6.101.3.23 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::LINE_TIME [static]

6.101.3.24 const MamaFieldDescriptor* Wombat::MamdaOrderImbalanceFields::SEND_TIME [static]
• MamdaOrderImbalanceFields.h
MamdaOrderImbalanceHandler is an interface for applications that want to have an easy way to handle order imbalance updates.

```cpp
#include <MamdaOrderImbalanceHandler.h>
```

**Public Member Functions**

- virtual void `onOrderImbalance (MamdaSubscription *subscription, MamdaOrderImbalanceListener &listener, const MamaMsg &msg, MamdaOrderImbalanceRecap &imbalance, MamdaOrderImbalanceUpdate &update)=0`
  Method invoked when an order imbalance is reported.

- virtual void `onNoOrderImbalance (MamdaSubscription *subscription, MamdaOrderImbalanceListener &listener, const MamaMsg &msg, MamdaOrderImbalanceRecap &imbalance, MamdaOrderImbalanceUpdate &update)=0`
  Method invoked when a no order imbalance is reported.

- virtual void `onOrderImbalanceRecap (MamdaSubscription *subscription, MamdaOrderImbalanceListener &listener, const MamaMsg &msg, MamdaOrderImbalanceRecap &imbalance)=0`
  Method invoked when the current order imbalance information is available.

- virtual `~MamdaOrderImbalanceHandler ()`

**6.102.1 Detailed Description**

MamdaOrderImbalanceHandler is an interface for applications that want to have an easy way to handle order imbalance updates.

The interface defines callback methods for different types of order-imbalance events:

**6.102.2 Constructor & Destructor Documentation**

**6.102.2.1 virtual Wombat::MamdaOrderImbalanceHandler::~MamdaOrderImbalanceHandler () [virtual]**

93 {};

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.102.3 Member Function Documentation

6.102.3.1 virtual void Wombat::MamdaOrderImbalanceHandler::onOrderImbalance (MamdaSubscription * subscription, MamdaOrderImbalanceListener & listener, const MamaMsg & msg, MamdaOrderImbalanceRecap & imbalance, MamdaOrderImbalanceUpdate & update) [pure virtual]

Method invoked when a order imbalance is reported.

Parameters:

subscription The subscription which received this update.
listener The listener which invoked this callback.
msg The MamaMsg that triggered this invocation.
imbalance Provides access to imbalance details
update Provides access to update details

6.102.3.2 virtual void Wombat::MamdaOrderImbalanceHandler::onNoOrderImbalance (MamdaSubscription * subscription, MamdaOrderImbalanceListener & listener, const MamaMsg & msg, MamdaOrderImbalanceRecap & imbalance, MamdaOrderImbalanceUpdate & update) [pure virtual]

Method invoked when a no order imbalance is reported.

Parameters:

subscription The subscription which received this update.
listener The listener which invoked this callback.
msg The MamaMsg that triggered this invocation.
imbalance Provides access to imbalance details
update Provides access to update details

6.102.3.3 virtual void Wombat::MamdaOrderImbalanceHandler::onOrderImbalanceRecap (MamdaSubscription * subscription, MamdaOrderImbalanceListener & listener, const MamaMsg & msg, MamdaOrderImbalanceRecap & imbalance) [pure virtual]

Method invoked when the current order imbalance information is available.

The reason for the invocation may be any of the following:
Initial image.

Recap update (e.g., after server fault tolerant event or data quality event.)

After stale status removed.

**Parameters:**

- `subscription` The subscription which received this update
- `listener` The listener which invoked this callback
- `msg` The MamaMsg that triggered this invocation
- `imbalance` Provides access to imbalance details

The documentation for this class was generated from the following file:

- `MamdaOrderImbalanceHandler.h`
A `MamdaOrderImbalanceListener` is a class that specializes in handling an imbalance order updates.

```cpp
#include <MamdaOrderImbalanceListener.h>
```

Inheritance diagram for `Wombat::MamdaOrderImbalanceListener`:

```
Wombat::MamdaBasicEvent
Wombat::MamdaOrderImbalanceUpdate
Wombat::MamdaMsgListener
Wombat::MamdaOrderImbalanceRecap
Wombat::MamdaBasicRecap
Wombat::MamdaOrderImbalanceListener
```

**Public Member Functions**

- `MamdaOrderImbalanceListener()`
- `virtual ~MamdaOrderImbalanceListener()`
- `void addHandler (MamdaOrderImbalanceHandler *handler)`
  
  *Add a specialized order imbalance handler.*

- `MamaPrice & getHighIndicationPrice () const`
- `MamdaFieldState getHighIndicationPriceFieldState () const`
  
  *Get the myHighIndicationPrice field state.*

- `MamaPrice & getLowIndicationPrice () const`
- `MamdaFieldState getLowIndicationPriceFieldState () const`
  
  *Get the myLowIndicationPrice field state.*

- `MamaPrice & getImbalancePrice () const`
- `MamdaFieldState getImbalancePriceFieldState () const`
  
  *Get the myIndicationPrice field state.*

- `int64_t getBuyVolume () const`
- `MamdaFieldState getBuyVolumeFieldState () const`
  
  *Get the myBuyVolume field state.*

- `int64_t getSellVolume () const`
- `MamdaFieldState getSellVolumeFieldState () const`
  
  *Get the mySellVolume field state.*
• int64_t getMatchVolume () const
  MamaFieldState getMatchVolumeFieldState () const
  Get the myMatchVolume field state.

• const char * getImbalanceState () const
  MamaFieldState getImbalanceStateFieldState () const
  Get the mySecurityStatusQual field state.

• MamaPrice & getMatchPrice () const
  MamaFieldState getMatchPriceFieldState () const
  Get the myInsideMatchPrice field state.

• MamaPrice & getFarClearingPrice () const
  MamaFieldState getFarClearingPriceFieldState () const
  Get the myFarClearingPrice field state.

• MamaPrice & getNearClearingPrice () const
  MamaFieldState getNearClearingPriceFieldState () const
  Get the myNearClearingPrice field state.

• char getNoClearingPrice () const
  MamaFieldState getNoClearingPriceFieldState () const
  Get the myNoClearingPrice field state.

• char getPriceVarInd () const
  MamaFieldState getPriceVarIndFieldState () const
  Get the myPriceVarInd field state.

• char getCrossType () const
  MamaFieldState getCrossTypeFieldState () const
  Get the myCrossType field state.

• MamaDateTime & getEventTime () const
  MamaFieldState getEventTimeFieldState () const
  Get the myEventTime field state.

• mama_seqnum_t getEventSeqNum () const
  MamaFieldState getEventSeqNumFieldState () const
  Get the myEventSeqNum field state.

• MamaDateTime & getSrctime () const
Get the source time.

- MamdaFieldState getSrcTimeFieldState () const
  Get the source time field state.

- MamaDateTime & getActivityTime () const
  Get the activity time.

- MamdaFieldState getActivityTimeFieldState () const
  Get the activity time field state.

- mama_i32_t getMsgType () const
  MamdaFieldState getMsgTypeFieldState () const
  Get the myMsgType field state.

- const char * getIssueSymbol () const
  MamdaFieldState getIssueSymbolFieldState () const
  Get the myIssueSymbol field state.

- const char * getPartId () const
  MamdaFieldState getPartIdFieldState () const
  Get the myPartId field state.

- mama_seqnum_t getSeqNum () const
  MamdaFieldState getSeqNumFieldState () const
  Get the mySeqNum field state.

- const char * getSecurityStatusOrig () const
  MamdaFieldState getSecurityStatusOrigFieldState () const
  Get the mySecurityStatusOrig field state.

- MamaDateTime & getSecurityStatusTime () const
  MamdaFieldState getSecurityStatusTimeFieldState () const
  Get the mySecurityStatusTime field state.

- MamaDateTime & getAuctionTime () const
  MamdaFieldState getAuctionTimeFieldState () const
  Get the myAuctionTime field state.

- MamaDateTime & getLineTime () const
  Get the line time.

- MamdaFieldState getLineTimeFieldState () const
Get the line time of the update.

- MamaDateTime & getSendTime() const
  Get the send time.

- MamaFieldState getSendTimeFieldState() const
  Get the send time field state.

- const char * getSymbol() const
  Get the instruments string symbol.

- MamaFieldState getSymbolFieldState() const
  Get the string symbol field state.

- const MamaMsgQual & getMsgQual() const
  Get the message qualifier.

- MamaFieldState getMsgQualFieldState() const
  Get the message qualifier field state.

- void updateFieldStates()
- virtual void onMsg(MamdaSubscription *subscription, const MamaMsg &msg, short msgType)
  Implementation of MamdaListener interface.

### 6.103.1 Detailed Description

A **MamdaOrderImbalanceListener** is class that specializes in handling an imbalance order updates.

An imbalance order occurs when too many orders of a particular type - either buy, sell or limit - for listed securities and not enough of the other, matching orders are received by an exchange. Developers provide their own implementation of the **MamdaOrderImbalanceHandler** interface and will be delivered notifications for imbalance updates. An obvious application for this MAMDA class is any kind of tick capture application.

**MamdaOrderImbalanceListener** should initialize the **MamdaOrderImbalanceFields** class prior to receiving the first message by calling **MamdaOrderImbalanceFields::setDictionary()** with a valid dictionary object which contains OrderImbalance related fields.
6.103.2 Constructor & Destructor Documentation

6.103.2.1 Wombat::MamdaOrderImbalanceListener::MamdaOrderImbalanceListener ()

6.103.2.2 virtual Wombat::MamdaOrderImbalanceListener::~MamdaOrderImbalanceListener () [virtual]

6.103.3 Member Function Documentation

6.103.3.1 void Wombat::MamdaOrderImbalanceListener::addHandler (MamdaOrderImbalanceHandler * handler)

Add a specialized order imbalance handler.

Returns:

Returns the myHighIndicationPrice.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.2 MamaPrice& Wombat::MamdaOrderImbalanceListener::getHighIndicationPrice () const [virtual]

Returns:

Returns the myHighIndicationPrice.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.3 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getHighIndicationPriceFieldState () const [virtual]

Get the myHighIndicationPrice field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.4 MamaPrice& Wombat::MamdaOrderImbalanceListener::getLowIndicationPrice () const [virtual]

Returns:

Returns the myLowIndicationPrice.

Implements Wombat::MamdaOrderImbalanceRecap.
6.103.3.5  **MamdaFieldState** Wombat::MamdaOrderImbalanceListener::getLowIndicationPriceFieldState () const [virtual]

Get the myLowIndicationPrice field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaOrderImbalanceRecap**.

6.103.3.6  **MamaPrice&** Wombat::MamdaOrderImbalanceListener::getImbalancePrice () const [virtual]

**Returns:**

Returns the myIndicationPrice.

Implements **Wombat::MamdaOrderImbalanceRecap**.

6.103.3.7  **MamdaFieldState** Wombat::MamdaOrderImbalanceListener::getImbalancePriceFieldState () const [virtual]

Get the myIndicationPrice field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaOrderImbalanceRecap**.

6.103.3.8  **int64_t** Wombat::MamdaOrderImbalanceListener::getBuyVolume () const [virtual]

**Returns:**

Returns the myBuyVolume.

Implements **Wombat::MamdaOrderImbalanceRecap**.
6.103.3.9 **MamdaFieldState** Wombat::MamdaOrderImbalanceListener::getBuyVolumeFieldState () const [virtual]

Get the myBuyVolume field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaOrderImbalanceRecap**.

6.103.3.10 **int64_t** Wombat::MamdaOrderImbalanceListener::getSellVolume () const [virtual]

**Returns:**

Returns the mySellVolume.

Implements **Wombat::MamdaOrderImbalanceRecap**.

6.103.3.11 **MamdaFieldState** Wombat::MamdaOrderImbalanceListener::getSellVolumeFieldState () const [virtual]

Get the mySellVolume field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaOrderImbalanceRecap**.

6.103.3.12 **int64_t** Wombat::MamdaOrderImbalanceListener::getMatchVolume () const [virtual]

**Returns:**

Returns the myMatchVolume.

Implements **Wombat::MamdaOrderImbalanceRecap**.
6.103.3.13  **MamdaFieldState**  Wombat::MamdaOrderImbalanceListener::getMatchVolumeFieldState () const  
[virtual]

Get the myMatchVolume field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.14  **const char** ∗ Wombat::MamdaOrderImbalanceListener::getImbalanceState () const  
[virtual]

**Returns:**

Returns the mySecurityStatusQual.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.15  **MamdaFieldState**  Wombat::MamdaOrderImbalanceListener::getImbalanceStateFieldState () const  
[virtual]

Get the mySecurityStatusQual field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.16  **MamaPrice** & Wombat::MamdaOrderImbalanceListener::getMatchPrice () const  
[virtual]

**Returns:**

Returns the myInsideMatchPrice.

Implements Wombat::MamdaOrderImbalanceRecap.
6.103.3.17 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getMatchPriceFieldState () const [virtual]

Get the myInsideMatchPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.18 MamaPrice& Wombat::MamdaOrderImbalanceListener::getFarClearingPrice () const [virtual]

Returns:
Returns the myFarClearingPrice.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.19 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getFarClearingPriceFieldState () const [virtual]

Get the myFarClearingPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.20 MamaPrice& Wombat::MamdaOrderImbalanceListener::getNearClearingPrice () const [virtual]

Returns:
Returns the myNearClearingPrice.

Implements Wombat::MamdaOrderImbalanceRecap.
6.103.21  MamdaFieldState Wombat::MamdaOrderImbalanceListener::getNearClearingPriceFieldState () const  
            [virtual]

Get the myNearClearingPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.22  char Wombat::MamdaOrderImbalanceListener::getNoClearingPrice () const  [virtual]

Returns:
Returns the myNoClearingPrice.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.23  MamdaFieldState Wombat::MamdaOrderImbalanceListener::getNoClearingPriceFieldState () const  
            [virtual]

Get the myNoClearingPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.24  char Wombat::MamdaOrderImbalanceListener::getPriceVarInd () const  [virtual]

Returns:
Returns the myPriceVarInd.

Implements Wombat::MamdaOrderImbalanceRecap.
6.103.3.25  \textbf{MamdaFieldState} Wombat::MamdaOrderImbalanceListener::getPriceVarIndFieldState () const  
[virtual]

Get the myPriceVarInd field state.

\textbf{Returns:}

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.26  \textbf{char} Wombat::MamdaOrderImbalanceListener::getCrossType () const  
[virtual]

\textbf{Returns:}

Returns the myCrossType.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.27  \textbf{MamdaFieldState} Wombat::MamdaOrderImbalanceListener::getCrossTypeFieldState () const  
[virtual]

Get the myCrossType field state.

\textbf{Returns:}

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.28  \textbf{MamaDateTime} Wombat::MamdaOrderImbalanceListener::getEventTime () const  
[virtual]

\textbf{Returns:}

Returns the myEventTime.

Implements Wombat::MamdaOrderImbalanceRecap.
6.103 Wombat::MamdaOrderImbalanceListener Class Reference

6.103.3.29 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getEventTimeFieldState () const [virtual]

Get the myEventTime field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.30 mama_seqnum_t Wombat::MamdaOrderImbalanceListener::getEventSeqNum () const [virtual]

Returns:
Returns the myEventSeqNum.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.31 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getEventSeqNumFieldState () const [virtual]

Get the myEventSeqNum field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.32 MamaDateTime& Wombat::MamdaOrderImbalanceListener::getSrcTime () const [virtual]

Get the source time.

Returns:
Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.
6.103.3.33  **MamdaFieldState** Wombat::MamdaOrderImbalanceListener::getSrcTimeFieldState () const

[virtual]

Get the source time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.103.3.34  **MamaDateTime**& Wombat::MamdaOrderImbalanceListener::getActivityTime () const

[virtual]

Get the activity time.

**Returns:**

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements Wombat::MamdaBasicEvent.

6.103.3.35  **MamdaFieldState** Wombat::MamdaOrderImbalanceListener::getActivityTimeFieldState () const

[virtual]

Get the activity time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.103.3.36  **mama_i32_t** Wombat::MamdaOrderImbalanceListener::getMsgType () const

[virtual]

Returns:

Returns the myMsgType.

Implements Wombat::MamdaOrderImbalanceRecap.
6.103.3.37 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getMsgTypeFieldState () const [virtual]

Get the myMsgType field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.38 const char∗ Wombat::MamdaOrderImbalanceListener::getIssueSymbol () const [virtual]

Returns:
Returns the myIssueSymbol.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.39 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getIssueSymbolFieldState () const [virtual]

Get the myIssueSymbol field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.40 const char∗ Wombat::MamdaOrderImbalanceListener::getPartId () const [virtual]

Returns:
Returns the myPartId.

Implements Wombat::MamdaOrderImbalanceRecap.
6.103.3.41 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getPartIdFieldState () const [virtual]
Get the myPartId field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.42 mama_seqnum_t Wombat::MamdaOrderImbalanceListener::getSeqNum () const [virtual]
Returns:
Returns the mySeqNum.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.43 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getSeqNumFieldState () const [virtual]
Get the mySeqNum field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.44 const char* Wombat::MamdaOrderImbalanceListener::getSecurityStatusOrig () const [virtual]
Returns:
Returns the mySecurityStatusOrig.

Implements Wombat::MamdaOrderImbalanceRecap.
6.103 Wombat::MamdaOrderImbalanceListener Class Reference

6.103.3.45 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getSecurityStatusOrigFieldState () const [virtual]

Get the mySecurityStatusOrig field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.46 MamaDateTime& Wombat::MamdaOrderImbalanceListener::getSecurityStatusTime () const [virtual]

Returns:
Returns the mySecurityStatusTime.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.47 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getSecurityStatusTimeFieldState () const [virtual]

Get the mySecurityStatusTime field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.48 MamaDateTime& Wombat::MamdaOrderImbalanceListener::getAuctionTime () const [virtual]

Returns:
Returns the Auction Time.

Implements Wombat::MamdaOrderImbalanceRecap.
6.103.3.49  

**MamdaFieldState** Wombat::MamdaOrderImbalanceListener::getAuctionTimeFieldState () const  
[virtual]

Get the myAuctionTime field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaOrderImbalanceRecap.

6.103.3.50  

**MamaDateTime**& Wombat::MamdaOrderImbalanceListener::getLineTime () const  
[virtual]

Get the line time.

**Returns:**

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

6.103.3.51  

**MamdaFieldState** Wombat::MamdaOrderImbalanceListener::getLineTimeFieldState () const  
[virtual]

Get the line time of the update.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.103.3.52  

**MamaDateTime**& Wombat::MamdaOrderImbalanceListener::getSendTime () const  
[virtual]

Get the send time.
Returns:

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime()).

Implements Wombat::MamdaBasicEvent.

6.103.3.53 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getSendTimeFieldState () const [virtual]

Get the send time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.103.3.54 const char* Wombat::MamdaOrderImbalanceListener::getSymbol () const [virtual]

Get the instruments string symbol.

Returns:

Symbol. This is the "well-known" symbol for the security, including any symbology mapping performed by the publisher.

Implements Wombat::MamdaBasicEvent.

6.103.3.55 MamdaFieldState Wombat::MamdaOrderImbalanceListener::getSymbolFieldState () const [virtual]

Get the string symbol field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.103.3.56  const MamaMsgQual& Wombat::MamdaOrderImbalanceListener::getMsgQual () const
[virtual]

Get the message qualifier.

Returns:

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.103.3.57  MamdaFieldState Wombat::MamdaOrderImbalanceListener::getMsgQualFieldState () const
[virtual]

Get the message qualifier field state.

Returns:

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.103.3.58  void Wombat::MamdaOrderImbalanceListener::updateFieldStates ()

6.103.3.59  virtual void Wombat::MamdaOrderImbalanceListener::onMsg (MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [virtual]

Implementation of MamdaListener interface.

Implements Wombat::MamdaMsgListener.

The documentation for this class was generated from the following file:

•  MamdaOrderImbalanceListener.h
6.104 Wombat::MamdaOrderImbalanceRecap Class Reference

#include <MamdaOrderImbalanceRecap.h>

Inheritance diagram for Wombat::MamdaOrderImbalanceRecap:

```
Wombat::MamdaBasicRecap

Wombat::MamdaOrderImbalanceRecap

Wombat::MamdaOrderImbalanceListener
```

Public Member Functions

- virtual const MamaDateTime & getAuctionTime () const =0
  
  Get the myAuctionTime field state.

- virtual MamdaFieldState getAuctionTimeFieldState () const =0

- virtual int64_t getBuyVolume () const =0
  
  Get the myBuyVolume field state.

- virtual MamdaFieldState getBuyVolumeFieldState () const =0

- virtual char getCrossType () const =0
  
  Get the myCrossType field state.

- virtual MamdaFieldState getCrossTypeFieldState () const =0

- virtual mama_seqnum_t getEventSeqNum () const =0
  
  Get the myEventSeqNum field state.

- virtual MamdaFieldState getEventSeqNumFieldState () const =0

- virtual const MamaDateTime & getEventTime () const =0
  
  Get the myEventTime field state.

- virtual MamdaFieldState getEventTimeFieldState () const =0

- virtual const MamaPrice & getFarClearingPrice () const =0
  
  Get the myFarClearingPrice field state.

- virtual MamdaFieldState getFarClearingPriceFieldState () const =0

- virtual const MamaPrice & getHighIndicationPrice () const =0

• virtual MamdaFieldState getHighIndicationPriceFieldState () const =0
  
  Get the myHighIndicationPrice field state.

• virtual const MamaPrice & getImbalancePrice () const =0
  • virtual MamdaFieldState getImbalancePriceFieldState () const =0
  
  Get the myIndicationPrice field state.

• virtual const MamaPrice & getMatchPrice () const =0
  • virtual MamdaFieldState getMatchPriceFieldState () const =0
  
  Get the myMatchPrice field state.

• virtual const char * getIssueSymbol () const =0
  • virtual MamdaFieldState getIssueSymbolFieldState () const =0
  
  Get the myIssueSymbol field state.

• virtual const MamaPrice & getLowIndicationPrice () const =0
  • virtual MamdaFieldState getLowIndicationPriceFieldState () const =0
  
  Get the myLowIndicationPrice field state.

• virtual int64_t getMatchVolume () const =0
  • virtual MamdaFieldState getMatchVolumeFieldState () const =0
  
  Get the myMatchVolume field state.

• virtual int32_t getMsgType () const =0
  • virtual MamdaFieldState getMsgTypeFieldState () const =0
  
  Get the myMsgType field state.

• virtual const MamaPrice & getNearClearingPrice () const =0
  • virtual MamdaFieldState getNearClearingPriceFieldState () const =0
  
  Get the myNearClearingPrice field state.

• virtual char getNoClearingPrice () const =0
  • virtual MamdaFieldState getNoClearingPriceFieldState () const =0
  
  Get the myNoClearingPrice field state.

• virtual const char * getPartId () const =0
  • virtual MamdaFieldState getPartIdFieldState () const =0
  
  Get the myPartId field state.

• virtual char getPriceVarInd () const =0
  • virtual MamdaFieldState getPriceVarIndFieldState () const =0
  
  Get the myPriceVarInd field state.
6.104 Wombat::MamdaOrderImbalanceRecap Class Reference

- virtual const char * getSecurityStatusOrig () const =0
- virtual MamdaFieldState getSecurityStatusOrigFieldState () const =0
  
  Get the mySecurityStatusOrig field state.

- virtual const char * getImbalanceState () const =0
- virtual MamdaFieldState getImbalanceStateFieldState () const =0
  
  Get the mySecurityStatusQual field state.

- virtual const MamaDateTime & getSecurityStatusTime () const =0
- virtual MamdaFieldState getSecurityStatusTimeFieldState () const =0
  
  Get the mySecurityStatusTime field state.

- virtual int64_t getSellVolume () const =0
- virtual MamdaFieldState getSellVolumeFieldState () const =0
  
  Get the mySellVolume field state.

- virtual mama_seqnum_t getSeqNum () const =0
- virtual MamdaFieldState getSeqNumFieldState () const =0
  
  Get the mySeqNum field state.

- virtual ~MamdaOrderImbalanceRecap ()

6.104.1 Constructor & Destructor Documentation

6.104.1.1 virtual Wombat::MamdaOrderImbalanceRecap::~MamdaOrderImbalanceRecap () [virtual]

303 {};

6.104.2 Member Function Documentation

6.104.2.1 virtual const MamaDateTime& Wombat::MamdaOrderImbalanceRecap::getAuctionTime () const [pure virtual]

Returns:

Returns the Auction Time.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.104.2.2 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getAuctionTimeFieldState () const [pure virtual]

Get the myAuctionTime field state.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.3 virtual int64_t Wombat::MamdaOrderImbalanceRecap::getBuyVolume () const [pure virtual]

**Returns:**
Returns the myBuyVolume.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.4 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getBuyVolumeFieldState () const [pure virtual]

Get the myBuyVolume field state.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.5 virtual char Wombat::MamdaOrderImbalanceRecap::getCrossType () const [pure virtual]

**Returns:**
Returns the myCrossType.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.104.2.6 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getCrossTypeFieldState () const [pure virtual]

Get the myCrossType field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.7 virtual mama_seqnum_t Wombat::MamdaOrderImbalanceRecap::getEventSeqNum () const [pure virtual]

Returns:

Returns the myEventSeqNum.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.8 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getEventSeqNumFieldState () const [pure virtual]

Get the myEventSeqNum field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.9 virtual const MamaDateTime& Wombat::MamdaOrderImbalanceRecap::getEventTime () const [pure virtual]

Returns:

Returns the myEventTime.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.104.2.10 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getEventTimeFieldState () const [pure virtual]

Get the myEventTime field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.11 virtual const MamaPrice& Wombat::MamdaOrderImbalanceRecap::getFarClearingPrice () const [pure virtual]

Returns:
Returns the myFarClearingPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.12 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getFarClearingPriceFieldState () const [pure virtual]

Get the myFarClearingPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.13 virtual const MamaPrice& Wombat::MamdaOrderImbalanceRecap::getHighIndicationPrice () const [pure virtual]

Returns:
Returns the myHighIndicationPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.104 Wombat::MamdaOrderImbalanceRecap Class Reference

6.104.2.14 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getHighIndicationPriceFieldState () const [pure virtual]

Get the myHighIndicationPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.15 virtual const MamaPrice& Wombat::MamdaOrderImbalanceRecap::getImbalancePrice () const [pure virtual]

Returns:
Returns the myIndicationPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.16 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getImbalancePriceFieldState () const [pure virtual]

Get the myIndicationPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.17 virtual const MamaPrice& Wombat::MamdaOrderImbalanceRecap::getMatchPrice () const [pure virtual]

Returns:
Returns the myInsideMatchPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.104.2.18 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getMatchPriceFieldState () const [pure virtual]

Get the myInsideMatchPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.19 virtual const char∗ Wombat::MamdaOrderImbalanceRecap::getIssueSymbol () const [pure virtual]

Returns:
Returns the myIssueSymbol.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.20 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getIssueSymbolFieldState () const [pure virtual]

Get the myIssueSymbol field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.21 virtual const MamaPrice& Wombat::MamdaOrderImbalanceRecap::getLowIndicationPrice () const [pure virtual]

Returns:
Returns the myLowIndicationPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.104 Wombat::MamdaOrderImbalanceRecap Class Reference

6.104.2.22 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getLowIndicationPriceFieldState () const [pure virtual]

Get the myLowIndicationPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.23 virtual int64_t Wombat::MamdaOrderImbalanceRecap::getMatchVolume () const [pure virtual]

Returns:
Returns the myMatchVolume.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.24 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getMatchVolumeFieldState () const [pure virtual]

Get the myMatchVolume field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.25 virtual int32_t Wombat::MamdaOrderImbalanceRecap::getMsgType () const [pure virtual]

Returns:
Returns the myMsgType.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.104.2.26 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getMsgTypeFieldState () const [pure virtual]

Get the myMsgType field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.27 virtual const MamaPrice& Wombat::MamdaOrderImbalanceRecap::getNearClearingPrice () const [pure virtual]

**Returns:**

Returns the myNearClearingPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.28 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getNearClearingPriceFieldState () const [pure virtual]

Get the myNearClearingPrice field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.29 virtual char Wombat::MamdaOrderImbalanceRecap::getNoClearingPrice () const [pure virtual]

**Returns:**

Returns the myNoClearingPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.104.2.30 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getNoClearingPriceFieldState () const [pure virtual]

Get the myNoClearingPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.31 virtual const char ∗ Wombat::MamdaOrderImbalanceRecap::getPartId () const [pure virtual]

Returns:
Returns the myPartId.

Implements Wombat::MamdaBasicRecap.
Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.32 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getPartIdFieldState () const [pure virtual]

Get the myPartId field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicRecap.
Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.33 virtual char Wombat::MamdaOrderImbalanceRecap::getPriceVarInd () const [pure virtual]

Returns:
Returns the myPriceVarInd.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.104.2.34 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getPriceVarIndFieldState () const [pure virtual]

Get the myPriceVarInd field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.35 virtual const char* Wombat::MamdaOrderImbalanceRecap::getSecurityStatusOrig () const [pure virtual]

**Returns:**

Returns the mySecurityStatusOrig.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.36 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getSecurityStatusOrigFieldState () const [pure virtual]

Get the mySecurityStatusOrig field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.37 virtual const char* Wombat::MamdaOrderImbalanceRecap::getImbalanceState () const [pure virtual]

**Returns:**

Returns the mySecurityStatusQual.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.104.2.38 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getImbalanceStateFieldState () const [pure virtual]

Get the mySecurityStatusQual field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.39 virtual const MamaDateTime& Wombat::MamdaOrderImbalanceRecap::getSecurityStatusTime () const [pure virtual]

**Returns:**

Returns the mySecurityStatusTime.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.40 virtual MamdaFieldState Wombat::MamdaOrderImbalanceRecap::getSecurityStatusTimeFieldState () const [pure virtual]

Get the mySecurityStatusTime field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.104.2.41 virtual int64_t Wombat::MamdaOrderImbalanceRecap::getSellVolume () const [pure virtual]

**Returns:**

Returns the mySellVolume.

Implemented in Wombat::MamdaOrderImbalanceListener.
Get the mySellVolume field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

Get the mySeqNum field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

The documentation for this class was generated from the following file:

- MamdaOrderImbalanceRecap.h
#include <MamdaOrderImbalanceSide.h>

## Public Types

- typedef enum Wombat::MamdaOrderImbalanceSide::OrderImbalanceSide_ OrderImbalanceSide
- ASK_SIDE_IMBALANCE_VALUE = 0
  
  ASK_SIDE_IMBALANCE.

- BUY_SIDE_IMBALANCE_VALUE
  
  BID_SIDE_IMBALANCE.

- NO_IMBALANCE_SIDE_VALUE
  
  NO_IMBALANCE.

- enum OrderImbalanceSide_ { ASK_SIDE_IMBALANCE_VALUE = 0, BUY_SIDE_IMBALANCE_VALUE, NO_IMBALANCE_SIDE_VALUE }

## Public Member Functions

- MamdaOrderImbalanceSide ()
- ~MamdaOrderImbalanceSide ()
- const char * valueToString (const int value)

## Member Typedef Documentation

### 6.105.1 Member Typedef Documentation

### 6.105.1.1 typedef enum Wombat::MamdaOrderImbalanceSide::OrderImbalanceSide_ Wombat::MamdaOrderImbalanceSide::OrderImbalanceSide

## Member Enumeration Documentation

### 6.105.2 Member Enumeration Documentation

### 6.105.2.1 enum Wombat::MamdaOrderImbalanceSide::OrderImbalanceSide_

**Enumerator:**

- ASK_SIDE_IMBALANCE_VALUE  ASK_SIDE_IMBALANCE.
- BUY_SIDE_IMBALANCE_VALUE  BID_SIDE_IMBALANCE.
NO_IMBALANCE_SIDE_VALUE  NO_IMBALANCE.

{  
   ASK_SIDE_IMBALANCE_VALUE = 0,
   BUY_SIDE_IMBALANCE_VALUE,
   NO_IMBALANCE_SIDE_VALUE

} OrderImbalanceSide;

6.105.3  Constructor & Destructor Documentation

6.105.3.1  Wombat::MamdaOrderImbalanceSide::MamdaOrderImbalanceSide ()

6.105.3.2  Wombat::MamdaOrderImbalanceSide::~MamdaOrderImbalanceSide ()

6.105.4  Member Function Documentation

6.105.4.1  const char* Wombat::MamdaOrderImbalanceSide::valueToString (const int value)

The documentation for this class was generated from the following file:

- MamdaOrderImbalanceSide.h
6.106 Wombat::MamdaOrderImbalanceType Class Reference

#include <MamdaOrderImbalanceType.h>

Public Types

- typedef enum Wombat::MamdaOrderImbalanceType::OrderImbalanceType::OrderImbalanceType_ OrderImbalanceType
- MARKET_IMBALANCE_BUY_VALUE = 16
  MARKET_IMBALANCE_BUY.
- MARKET_IMBALANCE_SELL_VALUE
  MARKET_IMBALANCE_SELL.
- NO_MARKET_IMBALANCE_VALUE
  NO_MARKET_IMBALANCE.
- MOC_IMBALANCE_BUY_VALUE
  MOC_IMBALANCE_BUY.
- MOC_IMBALANCE_SELL_VALUE
  MOC_IMBALANCE_SELL.
- NO_MOC_IMBALANCE_VALUE
  NO_MOC_IMBALANCE.
- ORDER_IMB_VALUE
  ORDER_IMB.
- ORDER_INF_VALUE
  ORDER_INF.
- ORDER_IMBALANCE_BUY_VALUE
  ORDER_IMBALANCE_BUY_VALUE.
- ORDER_IMBALANCE_SELL_VALUE
  ORDER_IMBALANCE_SELL_VALUE.
- NO_ORDER_IMBALANCE_VALUE
  NO_ORDER_IMBALANCE.
• **UNKNOWN** = -99

  
  UNKNOWN.

• enum OrderImbalanceType
  
  MARKET_IMBALANCE_BUY_VALUE = 16, MARKET_IMBALANCE_SELL_VALUE, NO_MARKET_IMBALANCE_VALUE, MOC_IMBALANCE_BUY_VALUE, MOC_IMBALANCE_SELL_VALUE, NO_MOC_IMBALANCE_VALUE, ORDER_IMB_VALUE, ORDER_INF_VALUE, ORDER_IMBALANCE_BUY_VALUE, ORDER_IMBALANCE_SELL_VALUE, NO_ORDER_IMBALANCE_VALUE, UNKNOWN = -99

Static Public Member Functions

• static const char ∗ valueToString (const int value)
• static const int stringToValue (const char ∗type)
• static bool isMamdaOrderImbalanceType (const int value)
  
  Determines if the value is of type order imbalance.

• static bool isMamdaImbalanceOrder (const int value)
  
  Determines if the value is of an order imbalance.

6.106.1 Member Typedef Documentation

6.106.1.1 typedef enum Wombat::MamdaOrderImbalanceType::OrderImbalanceType
  
  Wombat::MamdaOrderImbalanceType::OrderImbalanceType

6.106.2 Member Enumeration Documentation

6.106.2.1 enum Wombat::MamdaOrderImbalanceType::OrderImbalanceType

  Enumerator:

  MARKET_IMBALANCE_BUY_VALUE  MARKET_IMBALANCE_BUY.
  MARKET_IMBALANCE_SELL_VALUE  MARKET_IMBALANCE_SELL.
  NO_MARKET_IMBALANCE_VALUE  NO_MARKET_IMBALANCE.
  MOC_IMBALANCE_BUY_VALUE  MOC_IMBALANCE_BUY.
MOC_IMBALANCE_SELL_VALUE   MOC_IMBALANCE_SELL.

NO_MOC_IMBALANCE_VALUE  NO_MOC_IMBALANCE.

ORDER_IMB_VALUE  ORDER_IMB.

ORDER_INF_VALUE  ORDER_INF.

ORDER_IMBALANCE_BUY_VALUE  ORDER_IMBALANCE_BUY_VALUE.

ORDER_IMBALANCE_SELL_VALUE  ORDER_IMBALANCE_SELL_VALUE.

NO_ORDER_IMBALANCE_VALUE  NO_ORDER_IMBALANCE.

UNKNOWN  UNKNOWN.

{  
  MARKET_IMBALANCE_BUY_VALUE = 16,
  MARKET_IMBALANCE_SELL_VALUE,
  NO_MARKET_IMBALANCE_VALUE,
  MOC_IMBALANCE_BUY_VALUE,
  MOC_IMBALANCE_SELL_VALUE,
  NO_MOC_IMBALANCE_VALUE,
  ORDER_IMB_VALUE,
  ORDER_INF_VALUE,
  ORDER_IMBALANCE_BUY_VALUE,
  ORDER_IMBALANCE_SELL_VALUE,
  NO_ORDER_IMBALANCE_VALUE,
  UNKNOWN = -99
}

OrderImbalanceType;
6.106.3 Member Function Documentation

6.106.3.1 static const char* Wombat::MamdaOrderImbalanceType::valueToString (const int value) [static]

6.106.3.2 static const int Wombat::MamdaOrderImbalanceType::stringToValue (const char* type) [static]

6.106.3.3 static bool Wombat::MamdaOrderImbalanceType::isMamdaOrderImbalanceType (const int value) [static]

Determines if the value is of type order imbalance.

Parameters:

value

Returns:

bool

6.106.3.4 static bool Wombat::MamdaOrderImbalanceType::isMamdaImbalanceOrder (const int value) [static]

Determines if the value is of an order imbalance.

Parameters:

value

Returns:

bool

The documentation for this class was generated from the following file:

- MamdaOrderImbalanceType.h
6.107 Wombat::MamdaOrderImbalanceUpdate Class Reference

#include <MamdaOrderImbalanceUpdate.h>

Inheritance diagram for Wombat::MamdaOrderImbalanceUpdate::

```
Wombat::MamdaBasicEvent

Wombat::MamdaOrderImbalanceUpdate

Wombat::MamdaOrderImbalanceListener
```

Public Member Functions

- virtual const MamaDateTime & getAuctionTime () const =0
- virtual MamdaFieldState getAuctionTimeFieldState () const =0
  
  Get the myAuctionTime field state.

- virtual int64_t getBuyVolume () const =0
- virtual MamdaFieldState getBuyVolumeFieldState () const =0
  
  Get the myBuyVolume field state.

- virtual char getCrossType () const =0
- virtual MamdaFieldState getCrossTypeFieldState () const =0
  
  Get the myCrossType field state.

- virtual mama_seqnum_t getEventSeqNum () const =0
- virtual MamdaFieldState getEventSeqNumFieldState () const =0
  
  Get the myEventSeqNum field state.

- virtual const MamaDateTime & getEventTime () const =0
- virtual MamdaFieldState getEventTimeFieldState () const =0
  
  Get the myEventTime field state.

- virtual const MamaPrice & getFarClearingPrice () const =0
- virtual MamdaFieldState getFarClearingPriceFieldState () const =0
  
  Get the myFarClearingPrice field state.

- virtual const MamaPrice & getHighIndicationPrice () const =0
• virtual MamdaFieldState getHighIndicationPriceFieldState () const =0
  
  Get the myHighIndicationPrice field state.

• virtual const MamaPrice & getImbalancePrice () const =0
• virtual MamdaFieldState getImbalancePriceFieldState () const =0
  
  Get the myIndicationPrice field state.

• virtual const MamaPrice & getMatchPrice () const =0
• virtual MamdaFieldState getMatchPriceFieldState () const =0
  
  Get the myInsideMatchPrice field state.

• virtual const char * getIssueSymbol () const =0
• virtual MamdaFieldState getIssueSymbolFieldState () const =0
  
  Get the myIssueSymbol field state.

• virtual const MamaPrice & getLowIndicationPrice () const =0
• virtual MamdaFieldState getLowIndicationPriceFieldState () const =0
  
  Get the myLowIndicationPrice field state.

• virtual int64_t getMatchVolume () const =0
• virtual MamdaFieldState getMatchVolumeFieldState () const =0
  
  Get the myMatchVolume field state.

• virtual int32_t getMsgType () const =0
• virtual MamdaFieldState getMsgTypeFieldState () const =0
  
  Get the myMsgType field state.

• virtual const MamaPrice & getNearClearingPrice () const =0
• virtual MamdaFieldState getNearClearingPriceFieldState () const =0
  
  Get the myNearClearingPrice field state.

• virtual char getNoClearingPrice () const =0
• virtual MamdaFieldState getNoClearingPriceFieldState () const =0
  
  Get the myNoClearingPrice field state.

• virtual const char * getPartId () const =0
• virtual MamdaFieldState getPartIdFieldState () const =0
  
  Get the myPartId field state.

• virtual char getPriceVarInd () const =0
• virtual MamdaFieldState getPriceVarIndFieldState () const =0
  
  Get the myPriceVarInd field state.
• virtual const char ∗ getSecurityStatusOrig () const =0
• virtual MamdaFieldState getSecurityStatusOrigFieldState () const =0
  
  Get the mySecurityStatusOrig field state.

• virtual const char ∗ getImbalanceState () const =0
• virtual MamdaFieldState getImbalanceStateFieldState () const =0
  
  Get the mySecurityStatusQual field state.

• virtual const MamaDateTime & getSecurityStatusTime () const =0
• virtual MamdaFieldState getSecurityStatusTimeFieldState () const =0
  
  Get the mySecurityStatusTime field state.

• virtual int64_t getSellVolume () const =0
• virtual MamdaFieldState getSellVolumeFieldState () const =0
  
  Get the mySellVolume field state.

• virtual mama_seqnum_t getSeqNum () const =0
• virtual MamdaFieldState getSeqNumFieldState () const =0
  
  Get the mySeqNum field state.

• virtual ∼MamdaOrderImbalanceUpdate ()

6.107.1 Constructor & Destructor Documentation

6.107.1.1 virtual Wombat::MamdaOrderImbalanceUpdate::∼MamdaOrderImbalanceUpdate () [virtual]

304 {};

6.107.2 Member Function Documentation

6.107.2.1 virtual const MamaDateTime & Wombat::MamdaOrderImbalanceUpdate::getAuctionTime () const [pure
virtual]

Returns:

Returns the myAuctionTime.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.107.2.2 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getAuctionTimeFieldState () const [pure virtual]

Get the myAuctionTime field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.3 virtual int64_t Wombat::MamdaOrderImbalanceUpdate::getBuyVolume () const [pure virtual]

**Returns:**

Returns the myBuyVolume.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.4 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getBuyVolumeFieldState () const [pure virtual]

Get the myBuyVolume field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.5 virtual char Wombat::MamdaOrderImbalanceUpdate::getCrossType () const [pure virtual]

**Returns:**

Returns the myCrossType.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.107 Wombat::MamdaOrderImbalanceUpdate Class Reference

6.107.2.6 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getCrossTypeFieldState () const [pure virtual]

Get the myCrossType field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.7 virtual mama_seqnum_t Wombat::MamdaOrderImbalanceUpdate::getEventSeqNum () const [pure virtual]

Returns:
Returns the myEventSeqNum.

Implements Wombat::MamdaBasicEvent.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.8 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getEventSeqNumFieldState () const [pure virtual]

Get the myEventSeqNum field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.9 virtual const MamaDateTime& Wombat::MamdaOrderImbalanceUpdate::getEventTime () const [pure virtual]

Returns:
Returns the myEventTime.

Implements Wombat::MamdaBasicEvent.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.107.2.10 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getEventTimeFieldState () const [pure virtual]

Get the myEventTime field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.11 virtual const MamaPrice& Wombat::MamdaOrderImbalanceUpdate::getFarClearingPrice () const [pure virtual]

Returns:
Returns the myFarClearingPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.12 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getFarClearingPriceFieldState () const [pure virtual]

Get the myFarClearingPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.13 virtual const MamaPrice& Wombat::MamdaOrderImbalanceUpdate::getHighIndicationPrice () const [pure virtual]

Returns:
Returns the myHighIndicationPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.107.2.14 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getHighIndicationPriceFieldState () const [pure virtual]

Get the myHighIndicationPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.15 virtual const MamaPrice& Wombat::MamdaOrderImbalanceUpdate::getImbalancePrice () const [pure virtual]

Returns:
Returns the myIndicationPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.16 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getImbalancePriceFieldState () const [pure virtual]

Get the myIndicationPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.17 virtual const MamaPrice& Wombat::MamdaOrderImbalanceUpdate::getMatchPrice () const [pure virtual]

Returns:
Returns the myInsideMatchPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.
Get the myInsideMatchPrice field state.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaOrderImbalanceListener`.

Get the myIssueSymbol field state.

**Returns:**
Returns the myIssueSymbol.

Implemented in `Wombat::MamdaOrderImbalanceListener`.

Get the myIssueSymbol field state.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaOrderImbalanceListener`.

Returns the myLowIndicationPrice.

Implemented in `Wombat::MamdaOrderImbalanceListener`.
6.107.2.22 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getLowIndicationPriceFieldState () const [pure virtual]

Get the myLowIndicationPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.23 virtual int64_t Wombat::MamdaOrderImbalanceUpdate::getMatchVolume () const [pure virtual]

Returns:
Returns the myMatchVolume.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.24 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getMatchVolumeFieldState () const [pure virtual]

Get the myMatchVolume field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.25 virtual int32_t Wombat::MamdaOrderImbalanceUpdate::getMsgType () const [pure virtual]

Returns:
Returns the myMsgType.

Implemented in Wombat::MamdaOrderImbalanceListener.
Get the myMsgType field state.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

Get the myNearClearingPrice field state.

**Returns:**
Returns the myNearClearingPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.

Get the myNearClearingPrice field state.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

Returns the myNoClearingPrice.

Implemented in Wombat::MamdaOrderImbalanceListener.
virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getNoClearingPriceFieldState () const [pure virtual]

Get the myNoClearingPrice field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

virtual const char ∗ Wombat::MamdaOrderImbalanceUpdate::getPartId () const [pure virtual]

Returns:
Returns the myPartId.

Implements Wombat::MamdaBasicEvent.
Implemented in Wombat::MamdaOrderImbalanceListener.

virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getPartIdFieldState () const [pure virtual]

Get the myPartId field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaBasicEvent.
Implemented in Wombat::MamdaOrderImbalanceListener.

virtual char Wombat::MamdaOrderImbalanceUpdate::getPriceVarInd () const [pure virtual]

Returns:
Returns the myPriceVarInd.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.107.2.34 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getPriceVarIndFieldState () const [pure virtual]

Get the myPriceVarInd field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.35 virtual const char∗ Wombat::MamdaOrderImbalanceUpdate::getSecurityStatusOrig () const [pure virtual]

Returns:
Returns the mySecurityStatusOrig.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.36 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getSecurityStatusOrigFieldState () const [pure virtual]

Get the mySecurityStatusOrig field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.37 virtual const char∗ Wombat::MamdaOrderImbalanceUpdate::getImbalanceState () const [pure virtual]

Returns:
Returns the mySecurityStatusQual.

Implemented in Wombat::MamdaOrderImbalanceListener.
6.107.2.38 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getImbalanceStateFieldState () const [pure virtual]

Get the mySecurityStatusQual field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.39 virtual const MamaDateTime& Wombat::MamdaOrderImbalanceUpdate::getSecurityStatusTime () const [pure virtual]

Returns:
Returns the mySecurityStatusTime.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.40 virtual MamdaFieldState Wombat::MamdaOrderImbalanceUpdate::getSecurityStatusTimeFieldState () const [pure virtual]

Get the mySecurityStatusTime field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

6.107.2.41 virtual int64_t Wombat::MamdaOrderImbalanceUpdate::getSellVolume () const [pure virtual]

Returns:
Returns the mySellVolume.

Implemented in Wombat::MamdaOrderImbalanceListener.
Get the mySellVolume field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

Get the mySeqNum field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaOrderImbalanceListener.

The documentation for this class was generated from the following file:

* MamdaOrderImbalanceUpdate.h
6.108 Wombat::MamdaOrQuery Class Reference

#include <MamdaQuery.h>

Inheritance diagram for Wombat::MamdaOrQuery:

```
Wombat::MamdaQuery
```

Public Member Functions

- MamdaOrQuery (MamdaQuery *query1, MamdaQuery *query2)
- bool getXML (char *result)
- int getDepth ()
- void addQuery (MamdaQuery *query1)

6.108.1 Constructor & Destructor Documentation

6.108.1.1 Wombat::MamdaOrQuery::MamdaOrQuery (MamdaQuery *query1, MamdaQuery *query2)

6.108.2 Member Function Documentation

6.108.2.1 bool Wombat::MamdaOrQuery::getXML (char *result) [virtual]

Implements Wombat::MamdaQuery.

6.108.2.2 int Wombat::MamdaOrQuery::getDepth () [virtual]

Implements Wombat::MamdaQuery.

6.108.2.3 void Wombat::MamdaOrQuery::addQuery (MamdaQuery *query1)

The documentation for this class was generated from the following file:

- MamdaQuery.h
MamdaPubStatus is an interface that provides access to the Security Status fields such as symbol announce messages.

```cpp
#include <MamdaPubStatus.h>
```

Inheritance diagram for Wombat::MamdaPubStatus:

```
+-------------------+     +-------------------+
| Wombat::MamdaPubStatus | --> | Wombat::MamdaPubStatusListener |
```

Public Member Functions

- virtual const char * `getSymbol` () const =0
- virtual const char * `getActivityTime` () const =0
- virtual const char * `getActivityDate` () const =0
- virtual const char * `getPubId` () const =0
- virtual const char * `getPubClass` () const =0
- virtual const char * `getPubHostName` () const =0
- virtual const char * `getStatusMsgType` () const =0
- virtual const char * `getPubFtMode` () const =0
- virtual const char * `getPubMhMode` () const =0
- virtual const char * `getLineStatus` () const =0
- virtual long `getStatusSeqNum` () const =0
- virtual long `getBeginGapSeqNum` () const =0
- virtual long `getEndGapSeqNum` () const =0
- virtual long `getPubCacheSize` () const =0
- virtual const char * `getFreeText` () const =0
- virtual ~`MamdaPubStatus` ()

6.109.1 Detailed Description

MamdaPubStatus is an interface that provides access to the Security Status fields such as symbol announce messages.
6.109 Wombat::MamdaPubStatus Class Reference

6.109.2 Constructor & Destructor Documentation

6.109.2.1 virtual Wombat::MamdaPubStatus::~MamdaPubStatus ()
   [virtual]

54 {};

6.109.3 Member Function Documentation

6.109.3.1 virtual const char* Wombat::MamdaPubStatus::getSymbol () const
   [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.2 virtual const char* Wombat::MamdaPubStatus::getActivityTime () const
   [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.3 virtual const char* Wombat::MamdaPubStatus::getActivityDate () const
   [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.4 virtual const char* Wombat::MamdaPubStatus::getPubId () const
   [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.5 virtual const char* Wombat::MamdaPubStatus::getPubClass () const
   [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.6 virtual const char* Wombat::MamdaPubStatus::getPubHostName () const
   [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.
6.109.3.7 virtual const char* Wombat::MamdaPubStatus::getStatusMsgType () const [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.8 virtual const char* Wombat::MamdaPubStatus::getPubFtMode () const [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.9 virtual const char* Wombat::MamdaPubStatus::getPubMhMode () const [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.10 virtual const char* Wombat::MamdaPubStatus::getLineStatus () const [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.11 virtual long Wombat::MamdaPubStatus::getStatusSeqNum () const [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.12 virtual long Wombat::MamdaPubStatus::getBeginGapSeqNum () const [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.13 virtual long Wombat::MamdaPubStatus::getEndGapSeqNum () const [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

6.109.3.14 virtual long Wombat::MamdaPubStatus::getPubCacheSize () const [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.
6.109.3.15 virtual const char* Wombat::MamdaPubStatus::getFreeText ()
   const [pure virtual]

Implemented in Wombat::MamdaPubStatusListener.

The documentation for this class was generated from the following file:

- MamdaPubStatus.h
6.110 Wombat::MamdaPubStatusFields Class Reference

```cpp
#include <MamdaPubStatusFields.h>
```

**Static Public Member Functions**

- static void `setDictionary` (const MamaDictionary &dictionary)
- static void `reset` ()
  
  *Reset the dictionary for the update fields.*

- static uint16_t `getMaxFid` ()
- static bool `isSet` ()

**Static Public Attributes**

- static const MamaFieldDescriptor * `SYMBOL`
- static const MamaFieldDescriptor * `ACTIVITY_TIME`
- static const MamaFieldDescriptor * `ACTIVITY_DATE`
- static const MamaFieldDescriptor * `PUB_ID`
- static const MamaFieldDescriptor * `PUB_CLASS`
- static const MamaFieldDescriptor * `PUB_HOST_NAME`
- static const MamaFieldDescriptor * `STATUS_MSG_TYPE`
- static const MamaFieldDescriptor * `PUB_FT_MODE`
- static const MamaFieldDescriptor * `PUB_MH_MODE`
- static const MamaFieldDescriptor * `LINE_STATUS`
- static const MamaFieldDescriptor * `STATUS_SEQ_NUM`
- static const MamaFieldDescriptor * `MSG_GAP_BEGIN`
- static const MamaFieldDescriptor * `MSG_GAP_END`
- static const MamaFieldDescriptor * `PUB_CACHE_SIZE`
- static const MamaFieldDescriptor * `FREE_TEXT`

### 6.110.1 Member Function Documentation

#### 6.110.1.1 static void Wombat::MamdaPubStatusFields::setDictionary (const MamaDictionary &dictionary) [static]

#### 6.110.1.2 static void Wombat::MamdaPubStatusFields::reset () [static]

Reset the dictionary for the update fields.
6.110.1.3  static uint16_t Wombat::MamdaPubStatusFields::getMaxFid ()  
[static]

6.110.1.4  static bool Wombat::MamdaPubStatusFields::isSet ()  [static]

6.110.2  Member Data Documentation

6.110.2.1  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::SYMBOL  [static]

6.110.2.2  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::ACTIVITY_TIME  [static]

6.110.2.3  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::ACTIVITY_DATE  [static]

6.110.2.4  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::PUB_ID  [static]

6.110.2.5  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::PUB_CLASS  [static]

6.110.2.6  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::PUB_HOST_NAME  [static]

6.110.2.7  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::STATUS_MSG_TYPE  [static]

6.110.2.8  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::PUB_FT_MODE  [static]

6.110.2.9  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::PUB_MH_MODE  [static]

6.110.2.10  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::LINE_STATUS  [static]

6.110.2.11  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::STATUS_SEQ_NUM  [static]

6.110.2.12  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::MSG_GAP_BEGIN  [static]

6.110.2.13  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::MSG_GAP_END  [static]

6.110.2.14  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::PUB_CACHE_SIZE  [static]

6.110.2.15  const MamaFieldDescriptor* Wombat::MamdaPubStatusFields::FREE_TEXT  [static]

The documentation for this class was generated from the following file:
- MamdaPubStatusFields.h
6.111 Wombat::MamdaPubStatusHandler Class Reference

MamdaPubStatusHandler is an interface for applications that want to have an easy way to handle feed handler publisher status updates.

```cpp
#include <MamdaPubStatusHandler.h>
```

### Public Member Functions

- virtual void `onPubStatusUpdate (MamdaSubscription *subscription, MamdaPubStatusListener &listener, const MamaMsg &msg)=0`
  
  Method invoked when a quote is reported.

- virtual `~MamdaPubStatusHandler ()`

#### 6.111.1 Detailed Description

MamdaPubStatusHandler is an interface for applications that want to have an easy way to handle feed handler publisher status updates.

The interface defines callback methods for publisher status events:

#### 6.111.2 Constructor & Destructor Documentation

6.111.2.1 virtual Wombat::MamdaPubStatusHandler::~MamdaPubStatusHandler () [virtual]

55 {};

#### 6.111.3 Member Function Documentation

6.111.3.1 virtual void Wombat::MamdaPubStatusHandler::onPubStatusUpdate (MamdaSubscription * subscription, MamdaPubStatusListener & listener, const MamaMsg & msg) [pure virtual]

Method invoked when a quote is reported.

**Parameters:**

- `subscription` The MamdaSubscription handle.
- `listener` The listener which invoked this callback.
msg  The MamaMsg that triggered this invocation.

The documentation for this class was generated from the following file:

- MamdaPubStatusHandler.h
6.112 Wombat::MamdaPubStatusListener Class Reference

MamdaPubStatusListener is a class that specializes in handling Publisher (Feed Handler) Status updates.

#include <MamdaPubStatusListener.h>

Inheritance diagram for Wombat::MamdaPubStatusListener::

```
Wombat::MamdaMsgListener    Wombat::MamdaPubStatus
Wombat::MamdaPubStatusListener
```

Public Member Functions

- MamdaPubStatusListener()
- virtual ~MamdaPubStatusListener()
- void addHandler (MamdaPubStatusHandler *handler)
- void updateFieldStates()
- const char * getSymbol() const
  
  Accessor function for generic fields applicable to all 5 publisher status message types.

- const char * getActivityTime() const
- const char * getActivityDate() const
- const char * getPubId() const
- const char * getPubClass() const
- const char * getPubHostName() const
- MamdaFieldState getSymbolFieldState() const
- MamdaFieldState getActivityTimeFieldState() const
- MamdaFieldState getActivityDateFieldState() const
- MamdaFieldState getPubIdFieldState() const
- MamdaFieldState getPubClassFieldState() const
- MamdaFieldState getPubHostNameFieldState() const
- const char * getStatusMsgType() const
  
  Accessor function to the publisher status message type:
  - HB, FT, LS, MH, or GP.

- MamdaFieldState getStatusMsgTypeFieldState() const
- const char * getPubFtMode() const
Accessor functions for fields specific to a particular message type.

- const char * getPubMhMode () const
- const char * getLineStatus () const
- long getStatusSeqNum () const
- long getBeginGapSeqNum () const
- long getEndGapSeqNum () const
- long getPubCacheSize () const
- const char * getFreeText () const
- MamdaFieldState getPubFtModeFieldState () const
- MamdaFieldState getPubMhModeFieldState () const
- MamdaFieldState getLineStatusFieldState () const
- MamdaFieldState getStatusSeqNumFieldState () const
- MamdaFieldState getBeginGapSeqNumFieldState () const
- MamdaFieldState getEndGapSeqNumFieldState () const
- MamdaFieldState getPubCacheSizeFieldState () const
- MamdaFieldState getFreeTextFieldState () const
- virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)

Implementation of MamdaListener interface.

6.112.1 Detailed Description

MamdaPubStatusListener is a class that specializes in handling Publisher (Feed Handler) Status updates.

There are 5 types of publisher status msg:

- Heart Beat (StatusMsgType = "HB")
- Fault Tolerance (StatusMsgType = "FT")
- Line Status (StatusMsgType = "LS")
- Message Handler (StatusMsgType = "MH")
- Gap (StatusMsgType = "GP")

Note: The MamdaPubStatusListener class cache is cleared when a new update message arrives.

MamdaPubStatusListener should initialize the MamdaPubStatusFields class prior to receiving the first message by calling MamdaPubStatusFields::setDictionary() with a valid dictionary object which contains PubStatus related fields.
6.112.2 Constructor & Destructor Documentation

6.112.2.1 Wombat::MamdaPubStatusListener::MamdaPubStatusListener ()

6.112.2.2 virtual Wombat::MamdaPubStatusListener::~MamdaPubStatusListener () [virtual]

6.112.3 Member Function Documentation

6.112.3.1 void Wombat::MamdaPubStatusListener::addHandler (MamdaPubStatusHandler * handler)

6.112.3.2 void Wombat::MamdaPubStatusListener::updateFieldStates ()

6.112.3.3 const char * Wombat::MamdaPubStatusListener::getSymbol () const [virtual]

Accessor function for generic fields applicable to all 5 publisher status message types. Implements Wombat::MamdaPubStatus.

6.112.3.4 const char * Wombat::MamdaPubStatusListener::getActivityTime () const [virtual]

Implements Wombat::MamdaPubStatus.

6.112.3.5 const char * Wombat::MamdaPubStatusListener::getActivityDate () const [virtual]

Implements Wombat::MamdaPubStatus.

6.112.3.6 const char * Wombat::MamdaPubStatusListener::getPubId () const [virtual]

Implements Wombat::MamdaPubStatus.

6.112.3.7 const char * Wombat::MamdaPubStatusListener::getPubClass () const [virtual]

Implements Wombat::MamdaPubStatus.
6.112.3.8 const char* Wombat::MamdaPubStatusListener::getPubHostName () const [virtual]

Implements Wombat::MamdaPubStatus.

6.112.3.9 MamdaFieldState Wombat::MamdaPubStatusListener::getSymbolFieldState () const

6.112.3.10 MamdaFieldState Wombat::MamdaPubStatusListener::getActivityTimeFieldState () const

6.112.3.11 MamdaFieldState Wombat::MamdaPubStatusListener::getActivityDateFieldState () const

6.112.3.12 MamdaFieldState Wombat::MamdaPubStatusListener::getPubIdFieldState () const

6.112.3.13 MamdaFieldState Wombat::MamdaPubStatusListener::getPubClassFieldState () const

6.112.3.14 MamdaFieldState Wombat::MamdaPubStatusListener::getPubHostNameFieldState () const

6.112.3.15 const char* Wombat::MamdaPubStatusListener::getStatusMsgType () const [virtual]

Accessor function to the publisher status message type:

- HB, FT, LS, MH, or GP.

Implements Wombat::MamdaPubStatus.

6.112.3.16 MamdaFieldState Wombat::MamdaPubStatusListener::getStatusMsgTypeFieldState () const

6.112.3.17 const char* Wombat::MamdaPubStatusListener::getPubFtMode () const [virtual]

Accessor functions for fields specific to a particular message type.

The applicability of each accessor function to a given message type is defined in the comments below; e.g., "getPubFtMode" is only relevant and will only be populated for Heart Beat (HB) and Fault Tolerance (FT) publisher status messages.

Implements Wombat::MamdaPubStatus.
6.112.3.18 const char\* Wombat::MamdaPubStatusListener::getPubMhMode () const [virtual]

Implements Wombat::MamdaPubStatus.

6.112.3.19 const char\* Wombat::MamdaPubStatusListener::getLineStatus () const [virtual]

Implements Wombat::MamdaPubStatus.

6.112.3.20 long Wombat::MamdaPubStatusListener::getStatusSeqNum () const [virtual]

Implements Wombat::MamdaPubStatus.

6.112.3.21 long Wombat::MamdaPubStatusListener::getBeginGapSeqNum () const [virtual]

Implements Wombat::MamdaPubStatus.

6.112.3.22 long Wombat::MamdaPubStatusListener::getEndGapSeqNum () const [virtual]

Implements Wombat::MamdaPubStatus.

6.112.3.23 long Wombat::MamdaPubStatusListener::getPubCacheSize () const [virtual]

Implements Wombat::MamdaPubStatus.

6.112.3.24 const char\* Wombat::MamdaPubStatusListener::getFreeText () const [virtual]

Implements Wombat::MamdaPubStatus.
6.112.3.25 **MamdaFieldState** Wombat::MamdaPubStatusListener::getPubFmtModeFieldState () const

6.112.3.26 **MamdaFieldState** Wombat::MamdaPubStatusListener::getPubMhModeFieldState () const

6.112.3.27 **MamdaFieldState** Wombat::MamdaPubStatusListener::getLineStatusFieldState () const

6.112.3.28 **MamdaFieldState** Wombat::MamdaPubStatusListener::getStatusSeqNumFieldState () const

6.112.3.29 **MamdaFieldState** Wombat::MamdaPubStatusListener::getBeginGapSeqNumFieldState () const

6.112.3.30 **MamdaFieldState** Wombat::MamdaPubStatusListener::getEndGapSeqNumFieldState () const

6.112.3.31 **MamdaFieldState** Wombat::MamdaPubStatusListener::getPubCacheSizeFieldState () const

6.112.3.32 **MamdaFieldState** Wombat::MamdaPubStatusListener::getFreeTextFieldState () const

6.112.3.33 virtual void Wombat::MamdaPubStatusListener::onMsg (MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [virtual]

Implementation of MamdaListener interface.

Implements Wombat::MamdaMsgListener.

The documentation for this class was generated from the following file:

- MamdaPubStatusListener.h
6.113 Wombat::MamdaQualityListener Class Reference

MamdaQualityListener defines an interface for handling changes in quality notifications for a MamdaSubscription.

#include <MamdaQualityListener.h>

Public Member Functions

- virtual void onQuality (MamdaSubscription *subscription, mamaQuality quality)=0
- virtual ~MamdaQualityListener ()

6.113.1 Detailed Description

MamdaQualityListener defines an interface for handling changes in quality notifications for a MamdaSubscription.

6.113.2 Constructor & Destructor Documentation

6.113.2.1 virtual Wombat::MamdaQualityListener::~MamdaQualityListener () [virtual]

44 {};

6.113.3 Member Function Documentation

6.113.3.1 virtual void Wombat::MamdaQualityListener::onQuality (MamdaSubscription * subscription, mamaQuality quality) [pure virtual]

The documentation for this class was generated from the following file:

- MamdaQualityListener.h
#include <MamdaQuery.h>

Inheritance diagram for Wombat::MamdaQuery::

```
Wombat::MamdaQuery
Wombat::MamdaAndQuery
Wombat::MamdaContainsAllQuery
Wombat::MamdaContainsQuery
Wombat::MamdaDateQuery
Wombat::MamdaEqualsQuery
Wombat::MamdaOrQuery
```

Public Member Functions

- virtual bool getXML (char *result)=0
- bool getQuery (char *result)
- virtual int getDepth ()=0
- void setSubscriptionInfo (MamdaSubscription *subscInfo)
- MamdaSubscription * getSubscriptionInfo ()

Protected Member Functions

- MamdaQuery ()

6.114.1 Constructor & Destructor Documentation

6.114.1.1 Wombat::MamdaQuery::MamdaQuery () [protected]

6.114.2 Member Function Documentation

6.114.2.1 virtual bool Wombat::MamdaQuery::getXML (char * result) [pure virtual]

Implemented in Wombat::MamdaOrQuery, Wombat::MamdaAndQuery, Wombat::MamdaEqualsQuery, Wombat::MamdaDateQuery, Wombat::MamdaContainsAllQuery, and Wombat::MamdaContainsQuery.

6.114.2.2 bool Wombat::MamdaQuery::getQuery (char * & result)

6.114.2.3 virtual int Wombat::MamdaQuery::getDepth () [pure virtual]

Implemented in Wombat::MamdaOrQuery, Wombat::MamdaAndQuery, Wombat::MamdaEqualsQuery, Wombat::MamdaDateQuery, Wombat::MamdaContainsAllQuery, and Wombat::MamdaContainsQuery.

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.114.2.4  void Wombat::MamdaQuery::setSubscriptionInfo
           (MamdaSubscription * subscInfo)

6.114.2.5  MamdaSubscription* Wombat::MamdaQuery::getSubscriptionInfo
           ()

The documentation for this class was generated from the following file:

- MamdaQuery.h
MamdaQuoteChecker is a class that provides quotes sanity checking by periodically requesting snapshots of the quotes from the publisher and comparing that with an quotes being maintained in real time.

#include <MamdaQuoteChecker.h>

Public Member Functions

- MamdaQuoteChecker (MamaQueue *queue, MamdaCheckerHandler *handler, MamaSource *source, const char *symbol, mama_f64_t intervalSeconds)
- ~MamdaQuoteChecker ()
  Destructor.
- void checkSnapShotNow ()
  Perform an ad hoc snapshot check now.
- mama_u32_t getSuccessCount () const
- mama_u32_t getInconclusiveCount () const
  Return the number of inconclusive counts.
- mama_u32_t getFailureCount () const
  Return the number of failed checks.

6.115.1 Detailed Description

MamdaQuoteChecker is a class that provides quotes sanity checking by periodically requesting snapshots of the quotes from the publisher and comparing that with an quotes being maintained in real time.

This class is purely for testing purposes, to test for possible configuration or programming errors in the quotes publisher and in MAMDA quotes management code.

The developer registers a handler that contains callbacks for successful, inconclusive and failure events. The developer also provides an interval representing the frequency of the snapshot checks. The first check will take place at some random point in time between zero and the interval.
6.115.2 Constructor & Destructor Documentation

6.115.2.1 Wombat::MamdaQuoteChecker::MamdaQuoteChecker
(MamaQueue * queue, MamdaCheckerHandler * handler,
MamaSource * source, const char * symbol, mama_f64_t
intervalSeconds)

6.115.2.2 Wombat::MamdaQuoteChecker::~MamdaQuoteChecker ()

Destructor.

6.115.3 Member Function Documentation

6.115.3.1 void Wombat::MamdaQuoteChecker::checkSnapShotNow ()

Perform an ad hoc snapshot check now.
This may be useful if the checking is to be performed by some external trigger event.

6.115.3.2 mama_u32_t Wombat::MamdaQuoteChecker::getSuccessCount () const

Returns:
The number of successful checks.

6.115.3.3 mama_u32_t Wombat::MamdaQuoteChecker::getInconclusiveCount () const

Return the number of inconclusive counts.
An attempt to check the order book may be inconclusive if the order book sequence
numbers do not match up.

Returns:
The number of inconclusive checks.

6.115.3.4 mama_u32_t Wombat::MamdaQuoteChecker::getFailureCount () const

Return the number of failed checks.
This should be zero, of course.
Returns:

The number of failed checks.

The documentation for this class was generated from the following file:

- MamdaQuoteChecker.h
6.116  Wombat::MamdaQuoteClosing Class Reference

MamdaQuoteClosing is an interface that provides access to quote closing related fields.

#include <MamdaQuoteClosing.h>

Inheritance diagram for Wombat::MamdaQuoteClosing:

```
Wombat::MamdaBasicEvent
    Wombat::MamdaQuoteClosing
    Wombat::MamdaQuoteListener
```

Public Member Functions

- virtual const MamaPrice & getBidClosePrice () const =0
  Get the closing bid price.

- virtual MamdaFieldState getBidClosePriceFieldState () const =0
  Get the closing bid price field state.

- virtual const MamaPrice & getAskClosePrice () const =0
  Get the closing ask price.

- virtual MamdaFieldState getAskClosePriceFieldState () const =0
  Get the closing bid price field state.

- virtual ~MamdaQuoteClosing ()

6.116.1  Detailed Description

MamdaQuoteClosing is an interface that provides access to quote closing related fields.

6.116.2  Constructor & Destructor Documentation

6.116.2.1  virtual Wombat::MamdaQuoteClosing::~MamdaQuoteClosing ()
[virtual]

```
6.116.3 Member Function Documentation

6.116.3.1 virtual const MamaPrice& Wombat::MamdaQuoteClosing::getBidClosePrice () const [pure virtual]

Get the closing bid price.

**Returns:**

Today’s closing bid price, after the market has closed and the stock has traded today.

Implemented in Wombat::MamdaQuoteListener.

6.116.3.2 virtual MamdaFieldState Wombat::MamdaQuoteClosing::getBidClosePriceFieldState () const [pure virtual]

Get the closing bid price field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.116.3.3 virtual const MamaPrice& Wombat::MamdaQuoteClosing::getAskClosePrice () const [pure virtual]

Get the closing ask price.

**Returns:**

Today’s closing ask price, after the market has closed and the stock has traded today.

Implemented in Wombat::MamdaQuoteListener.

6.116.3.4 virtual MamdaFieldState Wombat::MamdaQuoteClosing::getAskClosePriceFieldState () const [pure virtual]

Get the closing bid price field state.
Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

The documentation for this class was generated from the following file:

- MamdaQuoteClosing.h
Utility cache of MamaFieldDescriptors which are used internally by the API when accessing quote related fields from update messages.

#include <MamdaQuoteFields.h>

Static Public Member Functions

- static void setDictionary (const MamaDictionary &dictionary)
- static void reset ()
  
  Reset the dictionary for quote update fields.

- static uint16_t getMaxFid ()
- static bool isSet ()

Static Public Attributes

- static const MamaFieldDescriptor * BID_PRICE
- static const MamaFieldDescriptor * BID_SIZE
- static const MamaFieldDescriptor * BID_DEPTH
- static const MamaFieldDescriptor * BID_PART_ID
- static const MamaFieldDescriptor * BID_CLOSE_PRICE
- static const MamaFieldDescriptor * BID_CLOSE_DATE
- static const MamaFieldDescriptor * BID_PREV_CLOSE_PRICE
- static const MamaFieldDescriptor * BID_PREV_CLOSE_DATE
- static const MamaFieldDescriptor * BID_HIGH
- static const MamaFieldDescriptor * BID_LOW
- static const MamaFieldDescriptor * ASK_PRICE
- static const MamaFieldDescriptor * ASK_SIZE
- static const MamaFieldDescriptor * ASK_DEPTH
- static const MamaFieldDescriptor * ASK_PART_ID
- static const MamaFieldDescriptor * ASK_CLOSE_PRICE
- static const MamaFieldDescriptor * ASK_CLOSE_DATE
- static const MamaFieldDescriptor * ASK_PREV_CLOSE_PRICE
- static const MamaFieldDescriptor * ASK_PREV_CLOSE_DATE
- static const MamaFieldDescriptor * ASK_HIGH
- static const MamaFieldDescriptor * ASK_LOW
- static const MamaFieldDescriptor * QUOTE_SEQ_NUM
- static const MamaFieldDescriptor * QUOTE_TIME
- static const MamaFieldDescriptor * QUOTE_DATE
- static const MamaFieldDescriptor * QUOTE_QUAL
• static const MamaFieldDescriptor * QUOTE_QUAL_NATIVE
• static const MamaFieldDescriptor * QUOTE_COUNT
• static const MamaFieldDescriptor * MID_PRICE
• static const MamaFieldDescriptor * SHORT_SALE_BID_TICK
• static const MamaFieldDescriptor * BID_TICK
• static const MamaFieldDescriptor * ASK_TIME
• static const MamaFieldDescriptor * BID_TIME
• static const MamaFieldDescriptor * ASK_INDICATOR
• static const MamaFieldDescriptor * BID_INDICATOR
• static const MamaFieldDescriptor * ASK_UPDATE_COUNT
• static const MamaFieldDescriptor * BID_UPDATE_COUNT
• static const MamaFieldDescriptor * ASK_YIELD
• static const MamaFieldDescriptor * BID_YIELD
• static const MamaFieldDescriptor * BID_SIZES_LIST
• static const MamaFieldDescriptor * ASK_SIZES_LIST
• static const MamaFieldDescriptor * BOOK_CONTRIBUTORS
• static const MamaFieldDescriptor * SHORT_SALE_CIRCUIT_BREAKER

6.117.1 Detailed Description

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing quote related fields from update messages.

This class should be initialized prior to using the MamaQuoteListener by calling setDictionary() with a valid dictionary object which contains quote related fields.

6.117.2 Member Function Documentation

6.117.2.1 static void Wombat::MamdaQuoteFields::setDictionary (const MamaDictionary & dictionary) [static]

6.117.2.2 static void Wombat::MamdaQuoteFields::reset () [static]

Reset the dictionary for quote update fields.
6.117.2.3 static uint16_t Wombat::MamdaQuoteFields::getMaxFid () [static]

6.117.2.4 static bool Wombat::MamdaQuoteFields::isSet () [static]

6.117.3 Member Data Documentation

6.117.3.1 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::BID_PRICE [static]

6.117.3.2 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::BID_SIZE [static]

6.117.3.3 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::BIDDEPTH [static]

6.117.3.4 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::BIDPART_ID [static]

6.117.3.5 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::BIDCLOSE_PRICE [static]

6.117.3.6 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::BIDCLOSE_DATE [static]

6.117.3.7 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::BIDPREV_CLOSE_PRICE [static]

6.117.3.8 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::BIDPREV_CLOSE_DATE [static]

6.117.3.9 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::BIDHIGH [static]

6.117.3.10 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::BIDLOW [static]

6.117.3.11 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::ASK_PRICE [static]

6.117.3.12 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::ASK_SIZE [static]

6.117.3.13 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::ASKDEPTH [static]

6.117.3.14 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::ASKPART_ID [static]

6.117.3.15 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::ASKCLOSE_PRICE [static]

6.117.3.16 const MamaFieldDescriptor* Wombat::MamdaQuoteFields::ASKCLOSE_DATE [static]
- MamdaQuoteFields.h
MamdaQuoteGap is an interface that provides access to quote gap related fields.

#include <MamdaQuoteGap.h>

Inheritance diagram for Wombat::MamdaQuoteGap::

```
Wombat::MamdaBasicEvent
  Wombat::MamdaQuoteGap
    Wombat::MamdaQuoteGap
    Wombat::MamdaBasicEvent
    Wombat::MamdaQuoteListener
```

Public Member Functions

- virtual mama_seqnum_t getBeginGapSeqNum () const =0
  
  The starting sequence number of detected missing quotes based on the quote count.

- virtual MamdaFieldState getBeginGapSeqNumFieldState () const =0
  
  Get the quote bid yield fieldState.

- virtual mama_seqnum_t getEndGapSeqNum () const =0
  
  The end sequence number of detected missing quotes based on the quote count.

- virtual MamdaFieldState getEndGapSeqNumFieldState () const =0
  
  Get the quote bid yield fieldState.

- virtual ~MamdaQuoteGap ()

6.118.1 Detailed Description

MamdaQuoteGap is an interface that provides access to quote gap related fields.

6.118.2 Constructor & Destructor Documentation

6.118.2.1 virtual Wombat::MamdaQuoteGap::~MamdaQuoteGap ()

[virtual]

69 {};

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.118 Wombat::MamdaQuoteGap Class Reference

6.118.3 Member Function Documentation

6.118.3.1 virtual mama_seqnum_t Wombat::MamdaQuoteGap::getBeginGapSeqNum () const [pure virtual]

The starting sequence number of detected missing quotes based on the quote count.

Returns:

The start gap sequence number.

Implemented in Wombat::MamdaQuoteListener.

6.118.3.2 virtual MamdaFieldState Wombat::MamdaQuoteGap::getBeginGapSeqNumFieldState () const [pure virtual]

Get the quote bid yield fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.118.3.3 virtual mama_seqnum_t Wombat::MamdaQuoteGap::getEndGapSeqNum () const [pure virtual]

The end sequence number of detected missing quotes based on the quote count.

Returns:

The end gap sequence number.

Implemented in Wombat::MamdaQuoteListener.

6.118.3.4 virtual MamdaFieldState Wombat::MamdaQuoteGap::getEndGapSeqNumFieldState () const [pure virtual]

Get the quote bid yield fieldState.
Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

The documentation for this class was generated from the following file:

- MamdaQuoteGap.h
MamdaQuoteHandler is an interface for applications that want to have an easy way to handle quote updates.

```cpp
#include <MamdaQuoteHandler.h>
```

### Public Member Functions

- **virtual void onQuoteRecap (MamdaSubscription *subscription, MamdaQuoteListener &listener, const MamaMsg &msg, const MamdaQuoteRecap &recap)=0**
  
  Method invoked when the current last-quote information for the security is available.

- **virtual void onQuoteUpdate (MamdaSubscription *subscription, MamdaQuoteListener &listener, const MamaMsg &msg, const MamdaQuoteUpdate &quote, const MamdaQuoteRecap &recap)=0**
  
  Method invoked when a quote is reported.

- **virtual void onQuoteGap (MamdaSubscription *subscription, MamdaQuoteListener &listener, const MamaMsg &msg, const MamdaQuoteGap &event, const MamdaQuoteRecap &recap)=0**
  
  Method invoked when a gap in quote reports is discovered.

- **virtual void onQuoteClosing (MamdaSubscription *subscription, MamdaQuoteListener &listener, const MamaMsg &msg, const MamdaQuoteClosing &event, const MamdaQuoteRecap &recap)=0**
  
  Method invoked for a closing report.

- **virtual void onQuoteOutOfSequence (MamdaSubscription *subscription, MamdaQuoteListener &listener, const MamaMsg &msg, const MamdaQuoteOutOfSequence &event, const MamdaQuoteRecap &recap)=0**
  
  Method invoked for a message marked as out of sequence.

- **virtual void onQuotePossiblyDuplicate (MamdaSubscription *subscription, MamdaQuoteListener &listener, const MamaMsg &msg, const MamdaQuotePossiblyDuplicate &event, const MamdaQuoteRecap &recap)=0**
  
  Method invoked for a message which is marked as possibly duplicate Listener must be configured to check the Msg Qualifier, i.e., call setControlProcessingByMsgQual on listener passing a value of true.

- **virtual ~MamdaQuoteHandler ()**
6.119.1 Detailed Description

MamdaQuoteHandler is an interface for applications that want to have an easy way to handle quote updates.

The interface defines callback methods for different types of quote-related events: quotes and closing-quote updates.

6.119.2 Constructor & Destructor Documentation

6.119.2.1 virtual Wombat::MamdaQuoteHandler::~MamdaQuoteHandler () [virtual]

149 {};

6.119.3 Member Function Documentation

6.119.3.1 virtual void Wombat::MamdaQuoteHandler::onQuoteRecap (MamdaSubscription * subscription, MamdaQuoteListener & listener, const MamaMsg & msg, const MamdaQuoteRecap & recap) [pure virtual]

Method invoked when the current last-quote information for the security is available.

The reason for the invocation may be any of the following:

- Initial image.
- Recap update (e.g., after server fault tolerant event or data quality event.)
- After stale status removed.

Parameters:

subscription The subscription which received the update.

listener The listener which invoked this callback.

msg The MamaMsg that triggered this invocation.

recap Access to the full quote recap details.

6.119.3.2 virtual void Wombat::MamdaQuoteHandler::onQuoteUpdate (MamdaSubscription * subscription, MamdaQuoteListener & listener, const MamaMsg & msg, const MamdaQuoteUpdate & quote, const MamdaQuoteRecap & recap) [pure virtual]

Method invoked when a quote is reported.
Parameters:

- **subscription** The subscription which received the update.
- **listener** The listener which invoked this callback.
- **msg** The MamaMsg that triggered this invocation.
- **quote** Access to the quote update details.
- **recap** Access to the full quote details.

**virtual void Wombat::MamdaQuoteHandler::onQuoteGap**

```
(MamdaSubscription * subscription, MamdaQuoteListener & listener, const MamaMsg & msg, const MamdaQuoteGap & event, const MamdaQuoteRecap & recap) [pure virtual]
```

Method invoked when a gap in quote reports is discovered.

Parameters:

- **subscription** The subscription which received the update.
- **listener** The listener which invoked this callback.
- **msg** The MamaMsg that triggered this invocation.
- **event** Access to the quote gap event details.
- **recap** Access to the full quote details.

**virtual void Wombat::MamdaQuoteHandler::onQuoteClosing**

```
(MamdaSubscription * subscription, MamdaQuoteListener & listener, const MamaMsg & msg, const MamdaQuoteClosing & event, const MamdaQuoteRecap & recap) [pure virtual]
```

Method invoked for a closing report.

Parameters:

- **subscription** The subscription which received the update.
- **listener** The listener which invoked this callback.
- **msg** The MamaMsg that triggered this invocation.
- **event** Access to the closing quote details.
- **recap** Access to the full quote details.
6.119.3.5 virtual void Wombat::MamdaQuoteHandler::onQuoteOutOfSequence (MamdaSubscription * subscription, MamdaQuoteListener & listener, const MamaMsg & msg, const MamdaQuoteOutOfSequence & event, const MamdaQuoteRecap & recap) [pure virtual]

Method invoked for a message marked as out of sequence.

Listener must be configured to check the Msg Qualifier, i.e., call setControlProcessingByMsgQual() on listener passing a value of true.

Parameters:

subscription The subscription which received the callback.
listener The quote listener which invoked this callback.
msg The MamaMsg that triggered this invocation.
event The possibly out of sequence event object.
recap The recap object.

6.119.3.6 virtual void Wombat::MamdaQuoteHandler::onQuotePossiblyDuplicate (MamdaSubscription * subscription, MamdaQuoteListener & listener, const MamaMsg & msg, const MamdaQuotePossiblyDuplicate & event, const MamdaQuoteRecap & recap) [pure virtual]

Method invoked for a message which is marked as possibly duplicate Listener must be configured to check the Msg Qualifier, i.e., call setControlProcessingByMsgQual on listener passing a value of true.

Parameters:

subscription The MamdaSubscription handle.
listener The quote listener.
msg The MamaMsg that triggered this invocation.
event The possibly duplicate event object.
recap The recap object.

The documentation for this class was generated from the following file:

- MamdaQuoteHandler.h
6.120 Wombat::MamdaQuoteListener Class Reference

MamdaQuoteListener is a class that specializes in handling quote updates.

#include <MamdaQuoteListener.h>

Inheritance diagram for Wombat::MamdaQuoteListener::

Public Member Functions

• MamdaQuoteListener ()
• virtual ~MamdaQuoteListener ()
• void addHandler (MamdaQuoteHandler *handler)
• void processPosDupAndOutOfSeqAsTransient (bool tf)
• void resolvePossiblyDuplicate (bool tf)
• void usePosDupAndOutOfSeqHandlers (bool tf)
• const char * getSymbol () const
  Get the instruments string symbol.

• const char * getPartId () const
  Get the participant identifier.

• const MamaDateTime & getSrcTime () const
  Get the source time.

• const MamaDateTime & getActivityTime () const
  Get the activity time.

• const MamaDateTime & getLineTime () const
  Get the line time.

• const MamaDateTime & getSendTime () const
  Get the send time.

• const MamaMsgQual & getMsgQual () const
  Get the message qualifier.

• const char * getPubId () const
• const MamaPrice & getBidPrice () const
  
  Get the quote bid price.

• mama_quantity_t getBidSize () const
  
  Get the quote bid size.

• mama_quantity_t getBidDepth () const
  
  Get the quote bid depth.

• const char * getBidPartId () const
  
  Get the quote bid participant identifier.

• const MamaPrice & getBidClosePrice () const
  
  Get the closing bid price.

• const MamaDateTime & getBidCloseDate () const
  
  Get the bid closing date.

• const MamaPrice & getBidPrevClosePrice () const
  
  Get the previous bid closing price.

• const MamaDateTime & getBidPrevCloseDate () const
  
  Get the previous bid closing date.

• const MamaPrice & getBidHigh () const
  
  Get the high bid price for the day.

• const MamaPrice & getBidLow () const
  
  Get the low bid price for the day.

• const MamaPrice & getAskPrice () const
  
  Get the quote ask price.

• mama_quantity_t getAskSize () const
  
  Get the quote ask size.

• mama_quantity_t getAskDepth () const
  
  Get the quote ask depth.

• const char * getAskPartId () const
  
  Get the quote ask participant identifier.
• const MamaPrice & getAskClosePrice () const
  Get the closing ask price.

• const MamaDateTime & getAskCloseDate () const
  Get the ask quote closing date.

• const MamaPrice & getAskPrevClosePrice () const
  Get the previous ask closing price.

• const MamaDateTime & getAskPrevCloseDate () const
  Get the previous ask closing date.

• const MamaPrice & getAskHigh () const
  Get the high ask price for the day.

• const MamaPrice & getAskLow () const
  Get the low ask price for the day.

• const MamaPrice & getQuoteMidPrice () const
  Get the quote mid price.

• mama_u32_t getQuoteCount () const
  Get the quote count.

• const char * getQuoteQualStr () const
  Get the quote qualifiers.

• const char * getQuoteQualNative () const
  Get the native feed quote qualifier.

• mama_seqnum_t getEventSeqNum () const
  Get the event sequence number.

• const MamaDateTime & getEventTime () const
  Get the event time.

• const MamaDateTime & getQuoteDate () const
  Get the quote date.

• mama_seqnum_t getBeginGapSeqNum () const
  The starting sequence number of detected missing quotes based on the quote count.
• mama_seqnum_t getEndGapSeqNum () const
  The end sequence number of detected missing quotes based on the quote count.

• char getShortSaleBidTick () const
  NASDAQ Bid Tick Indicator for Short Sale Rule Compliance.

• const MamaDateTime & getAskTime () const
  Get the quote ask time.

• const MamaDateTime & getBidTime () const
  Get the quote bid time.

• const char * getAskIndicator () const
  Get the quote ask indicator.

• const char * getBidIndicator () const
  Get the quote bid indicator.

• mama_u32_t getAskUpdateCount () const
  Get the quote ask update count.

• mama_u32_t getBidUpdateCount () const
  Get the quote bid update count.

• double getAskYield () const
  Get the quote ask yield.

• double getBidYield () const
  Get the quote bid yield.

• const char * getAskSizesList () const
  Get the ask sizes list.

• const char * getBidSizesList () const
  Get the bid sizes list.

• char getShortSaleCircuitBreaker () const
  get the ShortSaleCircuitBreaker

• MamdaFieldState getSymbolFieldState () const
  Get the string symbol field state.
• MamdaFieldState getBidPriceFieldState () const
  Get the quote bid yield field state.

• MamdaFieldState getBidSizeFieldState () const
  Get the quote bid yield field state.

• MamdaFieldState getPartIdFieldState () const
  Get the participant identifier field state.

• MamdaFieldState getSrcTimeFieldState () const
  Get the source time field state.

• MamdaFieldState getActivityTimeFieldState () const
  Get the activity time field state.

• MamdaFieldState getLineTimeFieldState () const
  Get the line time of the update.

• MamdaFieldState getSendTimeFieldState () const
  Get the send time field state.

• MamdaFieldState getPubIdFieldState () const
• MamdaFieldState getMsgQualFieldState () const
  Get the message qualifier field state.

• MamdaFieldState getPubIdModified () const
• MamdaFieldState getBidDepthFieldState () const
  Get the quote bid yield field state.

• MamdaFieldState getBidPartIdFieldState () const
  Get the quote bid yield field state.

• MamdaFieldState getBidClosePriceFieldState () const
  Get the closing bid price field state.

• MamdaFieldState getBidCloseDateFieldState () const
• MamdaFieldState getBidPrevClosePriceFieldState () const
• MamdaFieldState getBidPrevCloseDateFieldState () const
• MamdaFieldState getBidHighFieldState () const
• MamdaFieldState getBidLowFieldState () const
• MamdaFieldState getAskPriceFieldState () const
  Get the quote bid yield field state.
• MamdaFieldState getAskSizeFieldState () const
  Get the quote bid yield fieldState.

• MamdaFieldState getAskDepthFieldState () const
  Get the quote bid yield fieldState.

• MamdaFieldState getAskPartIdFieldState () const
  Get the quote bid yield fieldState.

• MamdaFieldState getAskClosePriceFieldState () const
  Get the closing bid price field state.

• MamdaFieldState getAskCloseDateFieldState () const
• MamdaFieldState getAskPrevClosePriceFieldState () const
• MamdaFieldState getAskPrevCloseDateFieldState () const
• MamdaFieldState getAskHighFieldState () const
• MamdaFieldState getAskLowFieldState () const
• MamdaFieldState getQuoteMidPriceFieldState () const
  Get the quote bid yield fieldState.

• MamdaFieldState getQuoteCountFieldState () const
• MamdaFieldState getQuoteQualStrFieldState () const
  Get the quote bid yield fieldState.

• MamdaFieldState getQuoteQualNativeFieldState () const
  Get the native feed quote qualifier fieldState.

• MamdaFieldState getEventSeqNumFieldState () const
  Get the event sequence number field state.

• MamdaFieldState getEventTypeFieldState () const
  Get the event time field state.

• MamdaFieldState getQuoteDateFieldState () const
  Get the quote date fieldState.

• MamdaFieldState getBeginGapSeqNumFieldState () const
  Get the quote bid yield fieldState.

• MamdaFieldState getEndGapSeqNumFieldState () const
  Get the quote bid yield fieldState.
• MamdaFieldState getShortSaleBidTickFieldState () const
  
  Get the short sale bid tick fieldState.

• MamdaFieldState getAskTimeFieldState () const
  
  Get the quote ask time fieldState.

• MamdaFieldState getBidTimeFieldState () const
  
  Get the quote bid time fieldState.

• MamdaFieldState getAskIndicatorFieldState () const
  
  Get the quote ask indicator fieldState.

• MamdaFieldState getBidIndicatorFieldState () const
  
  Get the quote bid indicator fieldState.

• MamdaFieldState getAskUpdateCountFieldState () const
  
  Get the quote ask update count fieldState.

• MamdaFieldState getBidUpdateCountFieldState () const
  
  Get the quote bid update count fieldState.

• MamdaFieldState getAskYieldFieldState () const
  
  Get the quote ask yield fieldState.

• MamdaFieldState getBidYieldFieldState () const
  
  Get the quote bid yield fieldState.

• MamdaFieldState getAskSizesListFieldState () const
  
  Get the ask sizes list fieldState.

• MamdaFieldState getBidSizesListFieldState () const
  
  Get the bid sizes list fieldState.

• MamdaFieldState getShortSaleCircuitBreakerFieldState () const

• virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)
  
  Implementation of MamdaListener interface.

• void assertEqual (MamdaQuoteListener *listener)
6.120.1 Detailed Description

MamdaQuoteListener is a class that specializes in handling quote updates.

Developers provide their own implementation of the MamdaQuoteHandler interface and will be delivered notifications for quotes and quote closing prices. An obvious application for this MAMDA class is any kind of quote tick capture application.

Note: The MamdaQuoteListener class caches quote-related field values. Among other reasons, caching of these fields makes it possible to provide complete quote-related callbacks, even when the publisher (e.g., feed handler) is only publishing deltas containing modified fields.

MamdaQuoteListener should initialize the MamdaQuoteFields class prior to receiving the first message by calling MamdaQuoteFields::setDictionary() with a valid dictionary object which contains Quote related fields.

6.120.2 Constructor & Destructor Documentation

6.120.2.1 Wombat::MamdaQuoteListener::MamdaQuoteListener()

6.120.2.2 virtual Wombat::MamdaQuoteListener::~MamdaQuoteListener()

6.120.3 Member Function Documentation

6.120.3.1 void Wombat::MamdaQuoteListener::addHandler(MamdaQuoteHandler * handler)

6.120.3.2 virtual Wombat::MamdaQuoteListener::processPosDupAndOutOfSeq-AsTransient(bool tf)

6.120.3.3 void Wombat::MamdaQuoteListener::resolvePossiblyDuplicate(bool tf)

6.120.3.4 void Wombat::MamdaQuoteListener::usePosDupAndOutOfSeq-Handlers(bool tf)

6.120.3.5 const char * Wombat::MamdaQuoteListener::getSymbol() const

Get the instruments string symbol.

Returns:

Symbol. This is the "well-known" symbol for the security, including any symbolology mapping performed by the publisher.
6.120 Wombat::MamdaQuoteListener Class Reference

Implements Wombat::MamdaBasicEvent.

6.120.3.6 const char* Wombat::MamdaQuoteListener::getPartId () const
[virtual]

Get the participant identifier.

Returns:
Participant ID. This may be an exchange identifier, a market maker ID, etc., or
NULL (if this is not related to any specific participant).

Implements Wombat::MamdaBasicEvent.

6.120.3.7 const MamaDateTime& Wombat::MamdaQuoteListener::getSrcTime () const
[virtual]

Get the source time.

Returns:
Source time. Typically, the exchange generated feed time stamp. This is often
the same as the "event time", because many feeds do not distinguish between the
actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.120.3.8 const MamaDateTime& Wombat::MamdaQuoteListener::getActivityTime () const
[virtual]

Get the activity time.

Returns:
Activity time. A feed handler generated time stamp representing when the data
item was last updated.

Implements Wombat::MamdaBasicEvent.

6.120.3.9 const MamaDateTime& Wombat::MamdaQuoteListener::getLineTime () const
[virtual]

Get the line time.
Returns:

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

6.120.3.10 const MamaDateTime& Wombat::MamdaQuoteListener::getSendTime () const [virtual]

Get the send time.

Returns:

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime()).

Implements Wombat::MamdaBasicEvent.

6.120.3.11 const MamaMsgQual& Wombat::MamdaQuoteListener::getMsgQual () const [virtual]

Get the message qualifier.

Returns:

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.120.3.12 const char* Wombat::MamdaQuoteListener::getPubId () const

6.120.3.13 const MamaPrice& Wombat::MamdaQuoteListener::getBidPrice () const [virtual]

Get the quote bid price.
Returns:

Bid price. The highest price that the representative party/group is willing to pay to buy the security. For most feeds, this size is represented in round lots.

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.14 mama_quantity_t Wombat::MamdaQuoteListener::getBidSize () const [virtual]

Get the quote bid size.

Returns:

Total share size available for the current bid price. Note: many feeds provide this size in terms of round lots.

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.15 mama_quantity_t Wombat::MamdaQuoteListener::getBidDepth () const [virtual]

Get the quote bid depth.

Returns:

The total size available at the current best bid price. Only supported by some exchanges. The total size may differ from the "best" size in that the total may be aggregated from multiple sources.

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.16 const char∗ Wombat::MamdaQuoteListener::getBidPartId () const [virtual]

Get the quote bid participant identifier.

Returns:

The identifier of the market participant (e.g. exchange or market maker) contributing the bid price field.

Implements Wombat::MamdaQuoteOutOfSequence.
6.120.3.17  \textbf{const MamaPrice& Wombat::MamdaQuoteListener::getBidClosePrice () const}  [virtual]

Get the closing bid price.

\textbf{Returns:}

Today's closing bid price, after the market has closed and the stock has traded today.

Implements  \texttt{Wombat::MamdaQuoteClosing}.

6.120.3.18  \textbf{const MamaDateTime& Wombat::MamdaQuoteListener::getBidCloseDate () const}  [virtual]

Get the bid closing date.

\textbf{Returns:}

Date of the BidClosePrice, if that value is non-zero. This is always the most recent day that the market was open, including today. This is different to BidPrevCloseDate.

Implements \texttt{Wombat::MamdaQuoteRecap}.

6.120.3.19  \textbf{const MamaPrice& Wombat::MamdaQuoteListener::getBidPrevClosePrice () const}  [virtual]

Get the previous bid closing price.

\textbf{Returns:}

The previous closing price, prior to any trade today.

Implements \texttt{Wombat::MamdaQuoteRecap}.

6.120.3.20  \textbf{const MamaDateTime& Wombat::MamdaQuoteListener::getBidPrevCloseDate () const}  [virtual]

Get the previous bid closing date.

\textbf{Returns:}

Date of the previous closing price, prior to any trade today.

Implements \texttt{Wombat::MamdaQuoteRecap}.
Get the high bid price for the day.

Returns:

High bid price.

Implements Wombat::MamdaQuoteRecap.

Get the low bid price for the day.

Returns:

Low bid price.

Implements Wombat::MamdaQuoteRecap.

Get the quote ask price.

Returns:

Ask price. The lowest price that the representative party/group is willing to take to sell the security. For most feeds, this size is represented in round lots.

Implements Wombat::MamdaQuoteOutOfSequence.

Get the quote ask size.

Returns:

Total share size available for the current ask price. Note: many feeds provide this size in terms of round lots.

Implements Wombat::MamdaQuoteOutOfSequence.
6.120.3.25 mama_quantity_t Wombat::MamdaQuoteListener::getAskDepth () const [virtual]

Get the quote ask depth.

Returns:

The total size available at the current best ask price. Only supported by some exchanges. The total size may differ from the "best" size in that the total may be aggregated from multiple sources.

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.26 const char∗ Wombat::MamdaQuoteListener::getAskPartId () const [virtual]

Get the quote ask participant identifier.

Returns:

The identifier of the market participant (e.g. exchange or market maker) contributing the ask price field.

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.27 const MamaPrice& Wombat::MamdaQuoteListener::getAskClosePrice () const [virtual]

Get the closing ask price.

Returns:

Today's closing ask price, after the market has closed and the stock has traded today.

Implements Wombat::MamdaQuoteClosing.

6.120.3.28 const MamaDateTime& Wombat::MamdaQuoteListener::getAskCloseDate () const [virtual]

Get the ask quote closing date.

Returns:

Date of the AskClosePrice, if that value is non-zero. This is always the most recent day that the market was open, including today. This is different to AskPrevCloseDate.
6.120 Wombat::MamdaQuoteListener Class Reference

Implements Wombat::MamdaQuoteRecap.

6.120.3.29  

\begin{verbatim}
const MamaPrice& Wombat::MamdaQuoteListener::getAskPrevClosePrice () const [virtual]
\end{verbatim}

Get the previous ask closing price.

**Returns:**

The previous closing price, prior to any trade today.

Implements Wombat::MamdaQuoteRecap.

6.120.3.30  

\begin{verbatim}
const MamaDateTime& Wombat::MamdaQuoteListener::getAskPrevCloseDate () const [virtual]
\end{verbatim}

Get the previous ask closing date.

**Returns:**

Date of previous closing price, prior to any trade today.

Implements Wombat::MamdaQuoteRecap.

6.120.3.31  

\begin{verbatim}
const MamaPrice& Wombat::MamdaQuoteListener::getAskHigh () const [virtual]
\end{verbatim}

Get the high ask price for the day.

**Returns:**

High ask price.

Implements Wombat::MamdaQuoteRecap.

6.120.3.32  

\begin{verbatim}
const MamaPrice& Wombat::MamdaQuoteListener::getAskLow () const [virtual]
\end{verbatim}

Get the low ask price for the day.

**Returns:**

Low ask price.

Implements Wombat::MamdaQuoteRecap.
6.120.3.33 const MamaPrice& Wombat::MamdaQuoteListener::getQuoteMidPrice () const [virtual]

Get the quote mid price.

Returns:

The mid price of the current quote. Usually, this is the average of the bid and ask prices, but some exchanges provide this field explicitly (e.g. LSE).

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.34 mama_u32_t Wombat::MamdaQuoteListener::getQuoteCount () const [virtual]

Get the quote count.

Returns:

The number of quotes generated for this security during the current trading session.

Implements Wombat::MamdaQuoteRecap.

6.120.3.35 const char* Wombat::MamdaQuoteListener::getQuoteQualStr () const [virtual]

Get the quote qualifiers.

Returns:

A normalized set of qualifiers for the last quote for the security. This field may contain multiple string values, separated by the colon(;) character.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Regular quote; no special condition</td>
</tr>
<tr>
<td>DepthAsk</td>
<td>Depth on ask side</td>
</tr>
<tr>
<td>DepthBid</td>
<td>Depth on bid side</td>
</tr>
<tr>
<td>Fast</td>
<td>Fast trading</td>
</tr>
<tr>
<td>NonFirm</td>
<td>Non-firm quote</td>
</tr>
<tr>
<td>Rotation</td>
<td>?</td>
</tr>
<tr>
<td>Auto</td>
<td>Automatic trade</td>
</tr>
<tr>
<td>Inactive</td>
<td>?</td>
</tr>
<tr>
<td>SpecBid</td>
<td>Specialist bid</td>
</tr>
<tr>
<td>SpecAsk</td>
<td>Specialist ask</td>
</tr>
<tr>
<td>Locked</td>
<td>Locked market - Bid is equal to Ask for OTCBB issues (Recaps only)</td>
</tr>
<tr>
<td>Crossed</td>
<td>Crossed market - Bid is greater than Ask for OTCBB (Recaps only)</td>
</tr>
<tr>
<td>Synd</td>
<td>Syndicate bid</td>
</tr>
<tr>
<td>PreSynd</td>
<td>Pre-syndicate bid</td>
</tr>
<tr>
<td>Penalty</td>
<td>Penalty bid</td>
</tr>
<tr>
<td>UnsolBid</td>
<td>Unsolicited bid</td>
</tr>
<tr>
<td>UnsolAsk</td>
<td>Unsolicited ask</td>
</tr>
<tr>
<td>UnsolQuote</td>
<td>Unsolicited quote</td>
</tr>
<tr>
<td>Firm</td>
<td>?</td>
</tr>
<tr>
<td>RangeInd</td>
<td>?</td>
</tr>
<tr>
<td>XpressBid</td>
<td>NYSE LiquidityQuote Xpress bid indicator</td>
</tr>
<tr>
<td>XpressAsk</td>
<td>NYSE LiquidityQuote Xpress ask indicator</td>
</tr>
</tbody>
</table>

Implements `Wombat::MamdaQuoteOutOfSequence`.

### 6.120.3.36 `const char* Wombat::MamdaQuoteListener::getQuoteQualNative () const [virtual]`

Get the native feed quote qualifier.

**Returns:**

Native quote qualifier (a.k.a. "quote condition"). Feed-specific quote qualifier code(s). This field is provided primarily for completeness and/or troubleshooting.

**See also:**

getQuoteQual.

Implements `Wombat::MamdaQuoteRecap`. 

---

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.120.3.37 mama_seqnum_t Wombat::MamdaQuoteListener::getEventSeqNum () const [virtual]

Get the event sequence number.

Returns:
Source sequence number. The exchange generated sequence number.

Implements Wombat::MamdaBasicEvent.

6.120.3.38 const MamaDateTime& Wombat::MamdaQuoteListener::getEventTime () const [virtual]

Get the event time.

Returns:
Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.120.3.39 const MamaDateTime& Wombat::MamdaQuoteListener::getQuoteDate () const [virtual]

Get the quote date.

Returns:
The quote date.

Implements Wombat::MamdaQuoteRecap.

6.120.3.40 mama_seqnum_t Wombat::MamdaQuoteListener::getBeginGapSeqNum () const [virtual]

The starting sequence number of detected missing quotes based on the quote count.

Returns:
The start gap sequence number.

Implements Wombat::MamdaQuoteGap.
6.120.3.41 `mama_seqnum_t Wombat::MamdaQuoteListener::getEndGapSeqNum () const` [virtual]

The end sequence number of detected missing quotes based on the quote count.

**Returns:**

The end gap sequence number.

Implements `Wombat::MamdaQuoteGap`.

6.120.3.42 `char Wombat::MamdaQuoteListener::getShortSaleBidTick () const` [virtual]

NASDAQ Bid Tick Indicator for Short Sale Rule Compliance.

National Bid Tick Indicator based on changes to the bid price of the National Best Bid or Offer (National BBO).

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>Up Tick. The current Best Bid Price price is higher than the previous Best Bid Price for the given NNM security.</td>
</tr>
<tr>
<td>D</td>
<td>Down Tick. The current Best Bid Price price is lower than the previous Best Bid Price for the given NNM security.</td>
</tr>
<tr>
<td>N</td>
<td>No Tick. The NASD Short Sale Rule does not apply to issue (i.e. NASDAQ SmallCap listed security).</td>
</tr>
<tr>
<td>Z</td>
<td>Unset - default value within the API</td>
</tr>
</tbody>
</table>

**Returns:**

The tick bid indicator.

Implements `Wombat::MamdaQuoteRecap`.

6.120.3.43 `const MamaDateTime& Wombat::MamdaQuoteListener::getAskTime () const` [virtual]

Get the quote ask time.

**Returns:**

The quote ask time.

Implements `Wombat::MamdaQuoteRecap`. 

---

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.120.3.44  const MamaDateTime& Wombat::MamdaQuoteListener::getBidTime () const  [virtual]

Get the quote bid time.

**Returns:**

The quote bid time.

Implements **Wombat::MamdaQuoteRecap**.

6.120.3.45  const char∗ Wombat::MamdaQuoteListener::getAskIndicator ()
const  [virtual]

Get the quote ask indicator.

**Returns:**

The quote ask indicator.

Implements **Wombat::MamdaQuoteRecap**.

6.120.3.46  const char∗ Wombat::MamdaQuoteListener::getBidIndicator ()
const  [virtual]

Get the quote bid indicator.

**Returns:**

The quote bid indicator.

Implements **Wombat::MamdaQuoteRecap**.

6.120.3.47  mama_u32_t Wombat::MamdaQuoteListener::getAskUpdateCount () const  [virtual]

Get the quote ask update count.

**Returns:**

The quote ask update count.

Implements **Wombat::MamdaQuoteRecap**.
Get the quote bid update count.

**Returns:**

The quote bid update count.

Implements Wombat::MamdaQuoteRecap.

Get the quote ask yield.

**Returns:**

The quote ask yield.

Implements Wombat::MamdaQuoteRecap.

Get the quote bid yield.

**Returns:**

The quote bid yield.

Implements Wombat::MamdaQuoteRecap.

Get the ask sizes list.

**Returns:**

The ask sizes list.

Implements Wombat::MamdaQuoteRecap.
6.120.3.52 const char* Wombat::MamdaQuoteListener::getBidSizesList () const [virtual]

Get the bid sizes list.

**Returns:**

The bid sizes list.

Implements Wombat::MamdaQuoteRecap.

6.120.3.53 char Wombat::MamdaQuoteListener::getShortSaleCircuitBreaker () const [virtual]

get the ShortSaleCircuitBreaker

**Returns:**

ShortSaleCircuitBreaker

- return values:
  - Blank: Short Sale Restriction Not in Effect.
  - A: Short Sale Restriction Activated.
  - C: Short Sale Restriction Continued.
  - D: Sale Restriction Deactivated.
  - E: Sale Restriction in Effect.

Implements Wombat::MamdaQuoteRecap.

6.120.3.54 MamdaFieldState Wombat::MamdaQuoteListener::getSymbolFieldState () const [virtual]

Get the string symbol field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.120.3.55 MamdaFieldState Wombat::MamdaQuoteListener::getBidPriceFieldState () const [virtual]

Get the quote bid yield field state.
Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.56 MamdaFieldState Wombat::MamdaQuoteListener::getBidSizeFieldState () const [virtual]

Get the quote bid yield field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.57 MamdaFieldState Wombat::MamdaQuoteListener::getPartIdFieldState () const [virtual]

Get the participant identifier field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.120.3.58 MamdaFieldState Wombat::MamdaQuoteListener::getSrcTimeFieldState () const [virtual]

Get the source time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.120.3.59 MamdaFieldState Wombat::MamdaQuoteListener::getActivityTimeFieldState () const [virtual]

Get the activity time field state.
Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.120.3.60 MamdaFieldState Wombat::MamdaQuoteListener::getLineTimeFieldState () const [virtual]

Get the line time of the update.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.120.3.61 MamdaFieldState Wombat::MamdaQuoteListener::getSendTimeFieldState () const [virtual]

Get the send time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.120.3.62 MamdaFieldState Wombat::MamdaQuoteListener::getPubIdFieldState () const
6.120.3.63 MamdaFieldState Wombat::MamdaQuoteListener::getMsgQualifierFieldState () const [virtual]

Get the message qualifier field state.

Returns:
Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.
Get the quote bid yield fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteOutOfSequence.

---

Get the quote bid yield fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteOutOfSequence.

---

Get the closing bid price field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteClosing.
MamdaFieldState Wombat::MamdaQuoteListener::getBidCloseDateFieldState () const

MamdaFieldState Wombat::MamdaQuoteListener::getBidPrevClosePriceFieldState () const

MamdaFieldState Wombat::MamdaQuoteListener::getBidPrevCloseDateFieldState () const

MamdaFieldState Wombat::MamdaQuoteListener::getBidHighFieldState () const

MamdaFieldState Wombat::MamdaQuoteListener::getBidLowFieldState () const

MamdaFieldState Wombat::MamdaQuoteListener::getAskPriceFieldState () const [virtual]

Get the quote bid yield fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteOutOfSequence.

MamdaFieldState Wombat::MamdaQuoteListener::getAskSizeFieldState () const [virtual]

Get the quote bid yield fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteOutOfSequence.

MamdaFieldState Wombat::MamdaQuoteListener::getAskDepthFieldState () const [virtual]

Get the quote bid yield fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.
6.120 Wombat::MamdaQuoteListener Class Reference

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.76 MamdaFieldState Wombat::MamdaQuoteListener::getAskPartIdFieldState () const [virtual]

Get the quote bid yield fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.77 MamdaFieldState Wombat::MamdaQuoteListener::getAskClosePriceFieldState () const [virtual]

Get the closing bid price field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteClosing.
6.120.3.78 MamdaFieldState Wombat::MamdaQuoteListener::getAskCloseDateFieldState () const

6.120.3.79 MamdaFieldState Wombat::MamdaQuoteListener::getAskPrevClosePriceFieldState () const

6.120.3.80 MamdaFieldState Wombat::MamdaQuoteListener::getAskPrevCloseDateFieldState () const

6.120.3.81 MamdaFieldState Wombat::MamdaQuoteListener::getAskHighFieldState () const

6.120.3.82 MamdaFieldState Wombat::MamdaQuoteListener::getAskLowFieldState () const

6.120.3.83 MamdaFieldState Wombat::MamdaQuoteListener::getQuoteMidPriceFieldState () const
   [virtual]

Get the quote bid yield fieldState.

Returns:
   MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteOutOfSequence.

6.120.3.84 MamdaFieldState Wombat::MamdaQuoteListener::getQuoteCountFieldState () const

6.120.3.85 MamdaFieldState Wombat::MamdaQuoteListener::getQuoteQualStrFieldState () const
   [virtual]

Get the quote bid yield fieldState.

Returns:
   MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteOutOfSequence.
6.120.3.86 MamdaFieldState Wombat::MamdaQuoteListener::getQuoteQualNativeFieldState () const 
[virtual]

Get the native feed quote qualifier fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.

6.120.3.87 MamdaFieldState Wombat::MamdaQuoteListener::getEventSeqNumFieldState () const 
[virtual]

Get the event sequence number field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.120.3.88 MamdaFieldState Wombat::MamdaQuoteListener::getEventTimeFieldState () const 
[virtual]

Get the event time field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.120.3.89 MamdaFieldState Wombat::MamdaQuoteListener::getQuoteDateFieldState () const 
[virtual]

Get the quote date fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.
Get the quote bid yield fieldState.

**Returns:**
- MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteGap.

Get the quote bid yield fieldState.

**Returns:**
- MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteGap.

Get the short sale bid tick fieldState.

**Returns:**
- MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.

Get the quote ask time fieldState.

**Returns:**
- MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.
Get the quote bid time fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.

Get the quote ask indicator fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.

Get the quote bid indicator fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.

Get the quote ask update count fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.
6.120.3.98 MamdaFieldState Wombat::MamdaQuoteListener::getBidUpdateCountFieldState () const [virtual]

Get the quote bid update count fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.

6.120.3.99 MamdaFieldState Wombat::MamdaQuoteListener::getAskYieldFieldState () const [virtual]

Get the quote ask yield fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.

6.120.3.100 MamdaFieldState Wombat::MamdaQuoteListener::getBidYieldFieldState () const [virtual]

Get the quote bid yield fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.

6.120.3.101 MamdaFieldState Wombat::MamdaQuoteListener::getAskSizesListFieldState () const [virtual]

Get the ask sizes list fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.
6.120.3.102  

**MamdaFieldState** Wombat::MamdaQuoteListener::getBidSizesListFieldState () const 

[virtual]

Get the bid sizes list fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.

6.120.3.103  

**MamdaFieldState** Wombat::MamdaQuoteListener::getShortSaleCircuitBreakerFieldState () const 

[virtual]

**Returns:**

The ShortSaleCircuitBreaker Field State. An enumeration representing field state.

Implements Wombat::MamdaQuoteRecap.

6.120.3.104  

**virtual void** Wombat::MamdaQuoteListener::onMsg (MamdaSubscription * subscription, const MamaMsg & msg, short msgType)  

[virtual]

Implementation of MamdaListener interface.

Implements Wombat::MamdaMsgListener.

6.120.3.105  

**void** Wombat::MamdaQuoteListener::assertEqual (MamdaQuoteListener * listener)

The documentation for this class was generated from the following file:

- MamdaQuoteListener.h
MamdaQuoteOutOfSequence is an interface that provides access to fields related to quote updates.

```cpp
#include <MamdaQuoteOutOfSequence.h>
```

Inheritance diagram for Wombat::MamdaQuoteOutOfSequence::

```
Wombat::MamdaBasicEvent
```

```
Wombat::MamdaQuoteOutOfSequence
```

```
Wombat::MamdaQuoteListener
```

### Public Member Functions

- virtual const MamaPrice & **getBidPrice** () const =0  
  *Get the quote bid price.*

- virtual MamdaFieldState **getBidPriceFieldState** () const =0  
  *Get the quote bid yield fieldState.*

- virtual mama_quantity_t **getBidSize** () const =0  
  *Get the quote bid size.*

- virtual MamdaFieldState **getBidSizeFieldState** () const =0  
  *Get the quote bid yield fieldState.*

- virtual mama_quantity_t **getBidDepth** () const =0  
  *Get the quote bid depth.*

- virtual MamdaFieldState **getBidDepthFieldState** () const =0  
  *Get the quote bid yield fieldState.*

- virtual const char ∗ **getBidPartId** () const =0  
  *Get the quote bid participant identifier.*

- virtual MamdaFieldState **getBidPartIdFieldState** () const =0  
  *Get the quote bid participant yield fieldState.*
Get the quote bid yield fieldState.

- virtual const MamaPrice & getAskPrice () const =0
  Get the quote ask price.

- virtual MamdaFieldState getAskPriceFieldState () const =0
  Get the quote bid yield fieldState.

- virtual mama_quantity_t getAskSize () const =0
  Get the quote ask size.

- virtual MamdaFieldState getAskSizeFieldState () const =0
  Get the quote bid yield fieldState.

- virtual mama_quantity_t getAskDepth () const =0
  Get the quote ask depth.

- virtual MamdaFieldState getAskDepthFieldState () const =0
  Get the quote bid yield fieldState.

- virtual const char * getAskPartId () const =0
  Get the quote ask participant identifier.

- virtual MamdaFieldState getAskPartIdFieldState () const =0
  Get the quote bid yield fieldState.

- virtual const MamaPrice & getQuoteMidPrice () const =0
  Get the quote mid price.

- virtual MamdaFieldState getQuoteMidPriceFieldState () const =0
  Get the quote bid yield fieldState.

- virtual const char * getQuoteQualStr () const =0
  Get the quote qualifiers.

- virtual MamdaFieldState getQuoteQualStrFieldState () const =0
  Get the quote bid yield fieldState.

- virtual ~MamdaQuoteOutOfSequence ()

### 6.121.1 Detailed Description

MamdaQuoteOutOfSequence is an interface that provides access to fields related to quote updates.
6.121.2 Constructor & Destructor Documentation

6.121.2.1 virtual Wombat::MamdaQuoteOutOfSequence::~MamdaQuoteOutOfSequence () [virtual]

294 {};

6.121.3 Member Function Documentation

6.121.3.1 virtual const MamaPrice& Wombat::MamdaQuoteOutOfSequence::getBidPrice () const [pure virtual]

Get the quote bid price.

Returns:

Bid price. The highest price that the representative party/group is willing to pay to buy the security. For most feeds, this size is represented in round lots.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.2 virtual MamaFieldState Wombat::MamdaQuoteOutOfSequence::getBidPriceFieldState () const [pure virtual]

Get the quote bid yield fieldState.

Returns:

MamaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.3 virtual mama_quantity_t Wombat::MamdaQuoteOutOfSequence::getBidSize () const [pure virtual]

Get the quote bid size.

Returns:

Total share size available for the current bid price. Note: many feeds provide this size in terms of round lots.

Implemented in Wombat::MamdaQuoteListener.
6.121.3.4 virtual MamdaFieldState Wombat::MamdaQuoteOutOfSequence::getBidSizeFieldState () const [pure virtual]

Get the quote bid yield fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.5 virtual mama_quantity_t Wombat::MamdaQuoteOutOfSequence::getBidDepth () const [pure virtual]

Get the quote bid depth.

Returns:
The total size available at the current best bid price. Only supported by some exchanges. The total size may differ from the "best" size in that the total may be aggregated from multiple sources.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.6 virtual MamdaFieldState Wombat::MamdaQuoteOutOfSequence::getBidDepthFieldState () const [pure virtual]

Get the quote bid yield fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.7 virtual const char* Wombat::MamdaQuoteOutOfSequence::getBidPartId () const [pure virtual]

Get the quote bid participant identifier.
6.121.3.8 virtual MamdaFieldState Wombat::MamdaQuoteOutOfSequence::getBidPartIdFieldState () const

Get the quote bid yield fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.9 virtual const MamaPrice& Wombat::MamdaQuoteOutOfSequence::getAskPrice () const

Get the quote ask price.

Returns:
Ask price. The lowest price that the representative party/group is willing to take to sell the security. For most feeds, this size is represented in round lots.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.10 virtual MamdaFieldState Wombat::MamdaQuoteOutOfSequence::getAskPriceFieldState () const

Get the quote bid yield fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
6.121 Wombat::MamdaQuoteOutOfSequence Class Reference

6.121.3.11 virtual mama_quantity_t Wombat::MamdaQuoteOutOfSequence::getAskSize () const  [pure virtual]

Get the quote ask size.

**Returns:**

Total share size available for the current ask price. Note: many feeds provide this size in terms of round lots.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.12 virtual MamdaFieldState Wombat::MamdaQuoteOutOfSequence::getAskSizeFieldState () const  [pure virtual]

Get the quote bid yield fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.13 virtual mama_quantity_t Wombat::MamdaQuoteOutOfSequence::getAskDepth () const  [pure virtual]

Get the quote ask depth.

**Returns:**

The total size available at the current best ask price. Only supported by some exchanges. The total size may differ from the "best" size in that the total may be aggregated from multiple sources.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.14 virtual MamdaFieldState Wombat::MamdaQuoteOutOfSequence::getAskDepthFieldState () const  [pure virtual]

Get the quote bid yield fieldState.
Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.15 virtual const char* Wombat::MamdaQuoteOutOfSequence::getAskPartId () const [pure virtual]

Get the quote ask participant identifier.

Returns:

The identifier of the market participant (e.g. exchange or market maker) contributing the ask price field.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.16 virtual MamdaFieldState Wombat::MamdaQuoteOutOfSequence::getAskPartIdFieldState () const [pure virtual]

Get the quote bid yield fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.17 virtual const MamaPrice& Wombat::MamdaQuoteOutOfSequence::getQuoteMidPrice () const [pure virtual]

Get the quote mid price.

Returns:

The mid price of the current quote. Usually, this is the average of the bid and ask prices, but some exchanges provide this field explicitly (e.g. LSE).

Implemented in Wombat::MamdaQuoteListener.
6.121.3.18 virtual MamdaFieldState Wombat::MamdaQuoteOutOfSequence::getQuoteMidPriceFieldState () const [pure virtual]

Get the quote bid yield fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.121.3.19 virtual const char∗ Wombat::MamdaQuoteOutOfSequence::getQuoteQualStr () const [pure virtual]

Get the quote qualifiers.

Returns:
A normalized set of qualifiers for the last quote for the security. This field may contain multiple string values, separated by the colon(:) character.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Regular quote; no special condition</td>
</tr>
<tr>
<td>DepthAsk</td>
<td>Depth on ask side</td>
</tr>
<tr>
<td>DepthBid</td>
<td>Depth on bid side</td>
</tr>
<tr>
<td>Fast</td>
<td>Fast trading</td>
</tr>
<tr>
<td>NonFirm</td>
<td>Non-firm quote</td>
</tr>
<tr>
<td>Rotation</td>
<td>?</td>
</tr>
<tr>
<td>Auto</td>
<td>Automatic trade</td>
</tr>
<tr>
<td>Inactive</td>
<td>?</td>
</tr>
<tr>
<td>SpecBid</td>
<td>Specialist bid</td>
</tr>
<tr>
<td>SpecAsk</td>
<td>Specialist ask</td>
</tr>
<tr>
<td>Locked</td>
<td>Locked market - Bid is equal to Ask for OTCBB issues (Recaps only)</td>
</tr>
<tr>
<td>Crossed</td>
<td>Crossed market - Bid is greater than Ask for OTCBB (Recaps only)</td>
</tr>
<tr>
<td>Synd</td>
<td>Syndicate bid</td>
</tr>
<tr>
<td>PreSynd</td>
<td>Pre-syndicate bid</td>
</tr>
<tr>
<td>Penalty</td>
<td>Penalty bid</td>
</tr>
<tr>
<td>UnsolBid</td>
<td>Unsolicited bid</td>
</tr>
<tr>
<td>UnsolAsk</td>
<td>Unsolicited ask</td>
</tr>
<tr>
<td>UnsolQuote</td>
<td>Unsolicited quote</td>
</tr>
<tr>
<td>Firm</td>
<td>?</td>
</tr>
<tr>
<td>RangeInd</td>
<td>?</td>
</tr>
<tr>
<td>XpressBid</td>
<td>NYSE LiquidityQuote Xpress bid indicator</td>
</tr>
<tr>
<td>XpressAsk</td>
<td>NYSE LiquidityQuote Xpress ask indicator</td>
</tr>
</tbody>
</table>

Implemented in Wombat::MamdaQuoteListener.

6.121.3.20 virtual MamdaFieldState Wombat::MamdaQuoteOutOfSequence::getQuoteQualStrFieldState () const [pure virtual]

Get the quote bid yield fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

The documentation for this class was generated from the following file:

- MamdaQuoteOutOfSequence.h
MamdaQuotePossiblyDuplicate is an interface that provides access to fields related to quote updates which are possible duplicates of previous quote updates.

```
#include <MamdaQuotePossiblyDuplicate.h>
```

Inheritance diagram for Wombat::MamdaQuotePossiblyDuplicate:

```
Wombat::MamdaBasicEvent
└── Wombat::MamdaQuotePossiblyDuplicate
    └── Wombat::MamdaQuoteListener
```

### Public Member Functions

- **virtual const MamaPrice & getBidPrice () const =0**
  
  *Get the quote bid price.*

- **virtual MamdaFieldState getBidPriceFieldState () const =0**
  
  *Get the field state.*

- **virtual mama_quantity_t getBidSize () const =0**
  
  *Get the quote bid size.*

- **virtual MamdaFieldState getBidSizeFieldState () const =0**
  
  *Get the field state.*

- **virtual mama_quantity_t getBidDepth () const =0**
  
  *Get the quote bid depth.*

- **virtual MamdaFieldState getBidDepthFieldState () const =0**
  
  *Get the field state.*

- **virtual const char * getBidPartId () const =0**
  
  *Get the quote bid participant identifier.*

- **virtual MamdaFieldState getBidPartIdFieldState () const =0**
  
  *Get the field state.*
Get the field state.

- virtual const MamaPrice & getAskPrice () const =0
  Get the quote ask price.

- virtual MamdaFieldState getAskPriceFieldState () const =0
  Get the field state.

- virtual mama_quantity_t getAskSize () const =0
  Get the quote ask size.

- virtual MamdaFieldState getAskSizeFieldState () const =0
  Get the field state.

- virtual mama_quantity_t getAskDepth () const =0
  Get the quote ask depth.

- virtual MamdaFieldState getAskDepthFieldState () const =0
  Get the field state.

- virtual const char ∗ getAskPartId () const =0
  Get the quote ask participant identifier.

- virtual MamdaFieldState getAskPartIdFieldState () const =0
  Get the field state.

- virtual const MamaPrice & getQuoteMidPrice () const =0
  Get the quote mid price.

- virtual MamdaFieldState getQuoteMidPriceFieldState () const =0
  Get the field state.

- virtual const char ∗ getQuoteQualStr () const =0
  Get the quote qualifier.

- virtual MamdaFieldState getQuoteQualStrFieldState () const =0
  Get the field state.

- virtual ~MamdaQuotePossiblyDuplicate ()

6.122.1 Detailed Description

MamdaQuotePossiblyDuplicate is an interface that provides access to fields related to quote updates which are possible duplicates of previous quote updates.
6.122 Wombat::MamdaQuotePossiblyDuplicate Class Reference

6.122.2 Constructor & Destructor Documentation

6.122.2.1 virtual Wombat::MamdaQuotePossiblyDuplicate::~MamdaQuotePossiblyDuplicate () [virtual]

293 {};

6.122.3 Member Function Documentation

6.122.3.1 virtual const MamaPrice& Wombat::MamdaQuotePossiblyDuplicate::getBidPrice () const [pure virtual]

Get the quote bid price.

Returns:
Bid price. The highest price that the representative party/group is willing to pay to buy the security. For most feeds, this size is represented in round lots.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.2 virtual MamaFieldState Wombat::MamdaQuotePossiblyDuplicate::getBidPriceFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.3 virtual mama_quantity_t Wombat::MamdaQuotePossiblyDuplicate::getBidSize () const [pure virtual]

Get the quote bid size.

Returns:
Total share size available for the current bid price. Note: many feeds provide this size in terms of round lots.

Implemented in Wombat::MamdaQuoteListener.
6.122.3.4 virtual MamdaFieldState Wombat::MamdaQuotePossiblyDuplicate::getBidSizeFieldState () const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.5 virtual mama_quantity_t Wombat::MamdaQuotePossiblyDuplicate::getBidDepth () const [pure virtual]

Get the quote bid depth.

**Returns:**

The total size available at the current best bid price. Only supported by some exchanges. The total size may differ from the "best" size in that the total may be aggregated from multiple sources.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.6 virtual MamdaFieldState Wombat::MamdaQuotePossiblyDuplicate::getBidDepthFieldState () const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.7 virtual const char* Wombat::MamdaQuotePossiblyDuplicate::getBidPartId () const [pure virtual]

Get the quote bid participant identifier.
Returns:

The identifier of the market participant (e.g. exchange or market maker) contributing the bid price field.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.8 virtual MamdaFieldState Wombat::MamdaQuotePossiblyDuplicate::getBidPartIdFieldState () const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.9 virtual const MamaPrice& Wombat::MamdaQuotePossiblyDuplicate::getAskPrice () const [pure virtual]

Get the quote ask price.

Returns:

Ask price. The lowest price that the representative party/group is willing to take to sell the security. For most feeds, this size is represented in round lots.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.10 virtual MamdaFieldState Wombat::MamdaQuotePossiblyDuplicate::getAskPriceFieldState () const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
6.122.3.11 virtual mama_quantity_t Wombat::MamdaQuotePossiblyDuplicate::getAskSize() const [pure virtual]

Get the quote ask size.

**Returns:**

Total share size available for the current ask price. Note: many feeds provide this size in terms of round lots.

Implemented in `Wombat::MamdaQuoteListener`.

6.122.3.12 virtual MamdaFieldState Wombat::MamdaQuotePossiblyDuplicate::getAskSizeFieldState() const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaQuoteListener`.

6.122.3.13 virtual mama_quantity_t Wombat::MamdaQuotePossiblyDuplicate::getAskDepth() const [pure virtual]

Get the quote ask depth.

**Returns:**

The total size available at the current best ask price. Only supported by some exchanges. The total size may differ from the "best" size in that the total may be aggregated from multiple sources.

Implemented in `Wombat::MamdaQuoteListener`.

6.122.3.14 virtual MamdaFieldState Wombat::MamdaQuotePossiblyDuplicate::getAskDepthFieldState() const [pure virtual]

Get the field state.
Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.15 virtual const char* Wombat::MamdaQuotePossiblyDuplicate::getAskPartId () const [pure virtual]

Get the quote ask participant identifier.

Returns:

The identifier of the market participant (e.g. exchange or market maker) contributing the ask price field.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.16 virtual MamdaFieldState Wombat::MamdaQuotePossiblyDuplicate::getAskPartIdFieldState () const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.122.3.17 virtual const MamaPrice& Wombat::MamdaQuotePossiblyDuplicate::getQuoteMidPrice () const [pure virtual]

Get the quote mid price.

Returns:

The mid price of the current quote. Usually, this is the average of the bid and ask prices, but some exchanges provide this field explicitly (e.g. LSE).

Implemented in Wombat::MamdaQuoteListener.
Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaQuoteListener`.

Get the quote qualifier.

**Returns:**

A normalized set of qualifiers for the last quote for the security. This field may contain multiple string values, separated by the colon character.
6.122 Wombat::MamdaQuotePossiblyDuplicate Class Reference

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Regular quote; no special condition</td>
</tr>
<tr>
<td>DepthAsk</td>
<td>Depth on ask side</td>
</tr>
<tr>
<td>DepthBid</td>
<td>Depth on bid side</td>
</tr>
<tr>
<td>Fast</td>
<td>Fast trading</td>
</tr>
<tr>
<td>NonFirm</td>
<td>Non-firm quote</td>
</tr>
<tr>
<td>Rotation</td>
<td>?</td>
</tr>
<tr>
<td>Auto</td>
<td>Automatic trade</td>
</tr>
<tr>
<td>Inactive</td>
<td>?</td>
</tr>
<tr>
<td>SpecBid</td>
<td>Specialist bid</td>
</tr>
<tr>
<td>SpecAsk</td>
<td>Specialist ask</td>
</tr>
<tr>
<td>Locked</td>
<td>Locked market - Bid is equal to Ask for OTCBB issues (Recaps only)</td>
</tr>
<tr>
<td>Crossed</td>
<td>Crossed market - Bid is greater than Ask for OTCBB (Recaps only)</td>
</tr>
<tr>
<td>Synd</td>
<td>Syndicate bid</td>
</tr>
<tr>
<td>PreSynd</td>
<td>Pre-syndicate bid</td>
</tr>
<tr>
<td>Penalty</td>
<td>Penalty bid</td>
</tr>
<tr>
<td>UnsolBid</td>
<td>Unsolicited bid</td>
</tr>
<tr>
<td>UnsolAsk</td>
<td>Unsolicited ask</td>
</tr>
<tr>
<td>UnsolQuote</td>
<td>Unsolicited quote</td>
</tr>
<tr>
<td>Firm</td>
<td>?</td>
</tr>
<tr>
<td>RangeInd</td>
<td>?</td>
</tr>
<tr>
<td>XpressBid</td>
<td>NYSE LiquidityQuote Xpress bid indicator</td>
</tr>
<tr>
<td>XpressAsk</td>
<td>NYSE LiquidityQuote Xpress ask indicator</td>
</tr>
</tbody>
</table>

Implemented in Wombat::MamdaQuoteListener.

6.122.3.20 virtual MamdaFieldState Wombat::MamdaQuotePossiblyDuplicate::getQuoteQualStrFieldState () const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

The documentation for this class was generated from the following file:

- MamdaQuotePossiblyDuplicate.h
MamdaQuoteRecap is an interface that provides access to quote related fields.

#include <MamdaQuoteRecap.h>

Inheritance diagram for Wombat::MamdaQuoteRecap:

```
Wombat::MamdaQuoteRecap
    Wombat::MamdaBasicRecap
    Wombat::MamdaQuoteListener
```

Public Member Functions

- virtual const MamaPrice & getBidPrice () const =0
  
  Get the quote bid price.

- virtual mama_quantity_t getBidSize () const =0
  
  Get the quote bid size.

- virtual mama_quantity_t getBidDepth () const =0
  
  Get the quote bid depth.

- virtual const char * getBidPartId () const =0
  
  Get the quote bid participant identifier.

- virtual const MamaPrice & getBidClosePrice () const =0
  
  Get the bid closing price.

- virtual const MamaDateTime & getBidCloseDate () const =0
  
  Get the bid closing date.

- virtual const MamaPrice & getBidPrevClosePrice () const =0
  
  Get the previous bid closing price.

- virtual const MamaDateTime & getBidPrevCloseDate () const =0
  
  Get the previous bid closing date.

- virtual const MamaPrice & getBidHigh () const =0
Get the high bid price for the day.

• virtual const MamaPrice & getBidLow () const =0
  Get the low bid price for the day.

• virtual const MamaPrice & getAskPrice () const =0
  Get the quote ask price.

• virtual mama_quantity_t getAskSize () const =0
  Get the quote ask size.

• virtual mama_quantity_t getAskDepth () const =0
  Get the quote ask depth.

• virtual const char * getAskPartId () const =0
  Get the quote ask participant identifier.

• virtual const MamaPrice & getAskClosePrice () const =0
  Get the ask quote closing price.

• virtual const MamaDateTime & getAskCloseDate () const =0
  Get the ask quote closing date.

• virtual const MamaPrice & getAskPrevClosePrice () const =0
  Get the previous ask closing price.

• virtual const MamaDateTime & getAskPrevCloseDate () const =0
  Get the previous ask closing date.

• virtual const MamaPrice & getAskHigh () const =0
  Get the high ask price for the day.

• virtual const MamaPrice & getAskLow () const =0
  Get the low ask price for the day.

• virtual const MamaPrice & getQuoteMidPrice () const =0
  Get the quote mid price.

• virtual mama_u32_t getQuoteCount () const =0
  Get the quote count.

• virtual const char * getQuoteQualStr () const =0
Get quote qualifier as a string.

• virtual const char * getQuoteQualNative () const =0
  Get the native feed quote qualifier.

• virtual char getShortSaleBidTick () const =0
  NASDAQ Bid Tick Indicator for Short Sale Rule Compliance.

• virtual const MamaDateTime & getAskTime () const =0
  Get the quote ask time.

• virtual const MamaDateTime & getBidTime () const =0
  Get the quote bid time.

• virtual const MamaDateTime & getQuoteDate () const =0
  Get the quote date.

• virtual const char * getAskIndicator () const =0
  Get the quote ask indicator.

• virtual const char * getBidIndicator () const =0
  Get the quote bid indicator.

• virtual mama_u32_t getAskUpdateCount () const =0
  Get the quote ask update count.

• virtual mama_u32_t getBidUpdateCount () const =0
  Get the quote bid update count.

• virtual double getAskYield () const =0
  Get the quote ask yield.

• virtual double getBidYield () const =0
  Get the quote bid yield.

• virtual const char * getAskSizesList () const =0
  Get the ask sizes list.

• virtual const char * getBidSizesList () const =0
  Get the bid sizes list.

• virtual char getShortSaleCircuitBreaker () const =0
get the `ShortSaleCircuitBreaker`

- virtual `MamdaFieldState getBidPriceFieldState () const =0`
  
  Get the quote bid price fieldState.

- virtual `MamdaFieldState getBidSizeFieldState () const =0`
  
  Get the quote bid size fieldState.

- virtual `MamdaFieldState getBidDepthFieldState () const =0`
  
  Get the quote bid depth fieldState.

- virtual `MamdaFieldState getBidPartIdFieldState () const =0`
  
  Get the quote participant identifier fieldState.

- virtual `MamdaFieldState getAskPriceFieldState () const =0`
  
  Get the quote ask price fieldState.

- virtual `MamdaFieldState getAskSizeFieldState () const =0`
  
  Get the quote ask size fieldState.

- virtual `MamdaFieldState getAskDepthFieldState () const =0`
  
  Get the quote ask depth fieldState.

- virtual `MamdaFieldState getAskPartIdFieldState () const =0`
  
  Get the quote ask participant identifier fieldState.

- virtual `MamdaFieldState getQuoteMidPriceFieldState () const =0`
  
  Get the quote mid price fieldState.

- virtual `MamdaFieldState getQuoteQualStrFieldState () const =0`
  
  Get the quote qualifier fieldState.

- virtual `MamdaFieldState getQuoteQualNativeFieldState () const =0`
  
  Get the native feed quote qualifier fieldState.

- virtual `MamdaFieldState getShortSaleBidTickFieldState () const =0`
  
  Get the short sale bid tick fieldState.

- virtual `MamdaFieldState getAskTimeFieldState () const =0`
  
  Get the quote ask time fieldState.

- virtual `MamdaFieldState getBidTimeFieldState () const =0`
Get the quote bid time fieldState.

- virtual MamdaFieldState getQuoteDateFieldState () const =0
  Get the quote date fieldState.

- virtual MamdaFieldState getAskIndicatorFieldState () const =0
  Get the quote ask indicator fieldState.

- virtual MamdaFieldState getBidIndicatorFieldState () const =0
  Get the quote bid indicator fieldState.

- virtual MamdaFieldState getAskUpdateCountFieldState () const =0
  Get the quote ask update count fieldState.

- virtual MamdaFieldState getBidUpdateCountFieldState () const =0
  Get the quote bid update count fieldState.

- virtual MamdaFieldState getAskYieldFieldState () const =0
  Get the quote ask yield fieldState.

- virtual MamdaFieldState getBidYieldFieldState () const =0
  Get the quote bid yield fieldState.

- virtual MamdaFieldState getAskSizesListFieldState () const =0
  Get the ask sizes list fieldState.

- virtual MamdaFieldState getBidSizesListFieldState () const =0
  Get the bid sizes list fieldState.

- virtual MamdaFieldState getShortSaleCircuitBreakerFieldState () const =0

6.123.1 Detailed Description

MamdaQuoteRecap is an interface that provides access to quote related fields.

6.123.2 Constructor & Destructor Documentation

6.123.2.1 virtual Wombat::MamdaQuoteRecap::~MamdaQuoteRecap ()
  [virtual]

710 {};

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.123 Wombat::MamdaQuoteRecap Class Reference

6.123.3 Member Function Documentation

6.123.3.1 virtual const MamaPrice& Wombat::MamdaQuoteRecap::getBidPrice () const [pure virtual]

Get the quote bid price.

Returns:
Bid price. The highest price that the representative party/group is willing to pay to buy the security. For most feeds, this size is represented in round lots.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.2 virtual mama_quantity_t Wombat::MamdaQuoteRecap::getBidSize () const [pure virtual]

Get the quote bid size.

Returns:
Total share size available for the current bid price. Note: many feeds provide this size in terms of round lots.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.3 virtual mama_quantity_t Wombat::MamdaQuoteRecap::getBidDepth () const [pure virtual]

Get the quote bid depth.

Returns:
The total size available at the current best bid price. Only supported by some exchanges. The total size may differ from the "best" size in that the total may be aggregated from multiple sources.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.4 virtual const char* Wombat::MamdaQuoteRecap::getBidPartId () const [pure virtual]

Get the quote bid participant identifier.
Returns:

The identifier of the market participant (e.g. exchange or market maker) contributing the bid price field.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.5 virtual const MamaPrice& Wombat::MamdaQuoteRecap::getBidClosePrice () const [pure virtual]

Get the bid closing price.

Returns:

Today's closing bid price, after the market has closed and the stock has traded today. If the market is not not closed or the stock did not trade today, zero is returned and the previous close price is available using getBidPrevClosePrice(). This value is always zero at the start of a trading day.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.6 virtual const MamaDateTime& Wombat::MamdaQuoteRecap::getBidCloseDate () const [pure virtual]

Get the bid closing date.

Returns:

Date of the BidClosePrice, if that value is non-zero. This is always the most recent day that the market was open, including today. This is different to BidPrevCloseDate.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.7 virtual const MamaPrice& Wombat::MamdaQuoteRecap::getBidPrevClosePrice () const [pure virtual]

Get the previous bid closing price.

Returns:

The previous closing price, prior to any trade today.

Implemented in Wombat::MamdaQuoteListener.
6.123.3.8 virtual const MamaDateTime& Wombat::MamdaQuoteRecap::getBidPrevCloseDate () const [pure virtual]

Get the previous bid closing date.

Returns:
Date of the previous closing price, prior to any trade today.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.9 virtual const MamaPrice& Wombat::MamdaQuoteRecap::getBidHigh () const [pure virtual]

Get the high bid price for the day.

Returns:
High bid price.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.10 virtual const MamaPrice& Wombat::MamdaQuoteRecap::getBidLow () const [pure virtual]

Get the low bid price for the day.

Returns:
Low bid price.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.11 virtual const MamaPrice& Wombat::MamdaQuoteRecap::getAskPrice () const [pure virtual]

Get the quote ask price.

Returns:
Ask price. The lowest price that the representative party/group is willing to take to sell the security. For most feeds, this size is represented in round lots.

Implemented in Wombat::MamdaQuoteListener.
6.123.3.12 virtual mama_quantity_t Wombat::MamdaQuoteRecap::getAskSize () const [pure virtual]

Get the quote ask size.

**Returns:**

Total share size available for the current ask price. Note: many feeds provide this size in terms of round lots.

Implemented in **Wombat::MamdaQuoteListener**.

6.123.3.13 virtual mama_quantity_t Wombat::MamdaQuoteRecap::getAskDepth () const [pure virtual]

Get the quote ask depth.

**Returns:**

The total size available at the current best ask price. Only supported by some exchanges. The total size may differ from the "best" size in that the total may be aggregated from multiple sources.

Implemented in **Wombat::MamdaQuoteListener**.

6.123.3.14 virtual const char * Wombat::MamdaQuoteRecap::getAskPartId () const [pure virtual]

Get the quote ask participant identifier.

**Returns:**

The identifier of the market participant (e.g. exchange or market maker) contributing the ask price field.

Implemented in **Wombat::MamdaQuoteListener**.

6.123.3.15 virtual const MamaPrice& Wombat::MamdaQuoteRecap::getAskClosePrice () const [pure virtual]

Get the ask quote closing price.
6.123 Wombat::MamdaQuoteRecap Class Reference

Returns:

Today’s closing ask price, after the market has closed and the stock has traded
today. If the market is not not closed or the stock did not trade today, zero is
returned and the previous close price is available using getAskPrevClose-
Price(). This value is always zero at the start of a trading day.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.16 virtual const MamaDateTime& Wombat::Mamda-
QuoteRecap::getAskCloseDate () const [pure
virtual]

Get the ask quote closing date.

Returns:

Date of the AskClosePrice, if that value is non-zero. This is always the most recent
day that the market was open, including today. This is different to AskPrevClose-
Date.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.17 virtual const MamaPrice& Wombat::MamdaQuote-
Recap::getAskPrevClosePrice () const [pure
virtual]

Get the previous ask closing price.

Returns:

The previous closing price, prior to any trade today.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.18 virtual const MamaDateTime& Wombat::Mamda-
QuoteRecap::getAskPrevCloseDate () const [pure
virtual]

Get the previous ask closing date.

Returns:

Date of previous closing price, prior to any trade today.

Implemented in Wombat::MamdaQuoteListener.
6.123.3.19 virtual const MamaPrice& Wombat::MamdaQuoteRecap::getAskHigh () const [pure virtual]

Get the high ask price for the day.

Returns:

High ask price.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.20 virtual const MamaPrice& Wombat::MamdaQuoteRecap::getAskLow () const [pure virtual]

Get the low ask price for the day.

Returns:

Low ask price.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.21 virtual const MamaPrice& Wombat::MamdaQuoteRecap::getQuoteMidPrice () const [pure virtual]

Get the quote mid price.

Returns:

The mid price of the current quote. Usually, this is the average of the bid and ask prices, but some exchanges provide this field explicitly (e.g. LSE).

Implemented in Wombat::MamdaQuoteListener.

6.123.3.22 virtual mama_u32_t Wombat::MamdaQuoteRecap::getQuoteCount () const [pure virtual]

Get the quote count.

Returns:

The number of quotes generated for this security during the current trading session.

Implemented in Wombat::MamdaQuoteListener.
Get quote qualifier as a string.

Returns:
A normalized set of qualifiers for the last quote for the security. This field may contain multiple string values, separated by the colon(:) character.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Regular quote; no special condition</td>
</tr>
<tr>
<td>DepthAsk</td>
<td>Depth on ask side</td>
</tr>
<tr>
<td>DepthBid</td>
<td>Depth on bid side</td>
</tr>
<tr>
<td>SlowQuoteOnAskSide</td>
<td>This indicates that a market participants Ask is in a slow (CTA) mode. While in this mode, automated execution is not eligible on the Ask side and can be traded through pursuant to Regulation NMS requirements.</td>
</tr>
<tr>
<td>SlowQuoteOnBidSide</td>
<td>This indicates that a market participants Bid is in a slow (CTA) mode. While in this mode, automated execution is not eligible on the Bid side and can be traded through pursuant to Regulation NMS requirements.</td>
</tr>
<tr>
<td>Fast</td>
<td>Fast trading</td>
</tr>
<tr>
<td>NonFirm</td>
<td>Non-firm quote</td>
</tr>
<tr>
<td>Rotation</td>
<td>OPRA only. Quote relates to a trading rotation (Where a participant rotates through various clients that they are trading for)</td>
</tr>
<tr>
<td>Auto</td>
<td>Automatic trade</td>
</tr>
<tr>
<td>Inactive</td>
<td></td>
</tr>
<tr>
<td>SpecBid</td>
<td>Specialist bid</td>
</tr>
<tr>
<td>SpecAsk</td>
<td>Specialist ask</td>
</tr>
<tr>
<td>OneSided</td>
<td>One sided. No orders, or only market orders, exist on one side of the book.</td>
</tr>
<tr>
<td>PassiveMarketMaker</td>
<td>Market Maker is both underwriter and buyer of security.</td>
</tr>
<tr>
<td>LockedMarket</td>
<td>Locked market - Bid is equal to Ask for OTCBB issues</td>
</tr>
<tr>
<td>Crossed</td>
<td>Crossed market - Bid is greater than Ask for OTCBB</td>
</tr>
<tr>
<td>Synd</td>
<td>Syndicate bid</td>
</tr>
<tr>
<td>PreSynd</td>
<td>Pre-syndicate bid</td>
</tr>
<tr>
<td>Penalty</td>
<td>Penalty bid</td>
</tr>
<tr>
<td>UnsolBid</td>
<td>Unsolicited bid</td>
</tr>
<tr>
<td>UnsolAsk</td>
<td>Unsolicited ask</td>
</tr>
<tr>
<td>UnsolQuote</td>
<td>Unsolicited quote</td>
</tr>
<tr>
<td>Empty</td>
<td>Empty quote (no quote)</td>
</tr>
<tr>
<td>XpressBid</td>
<td>NYSE LiquidityQuote Xpress bid indicator</td>
</tr>
<tr>
<td>XpressAsk</td>
<td>NYSE LiquidityQuote Xpress ask indicator</td>
</tr>
<tr>
<td>BestOrder</td>
<td></td>
</tr>
<tr>
<td>WillSell</td>
<td></td>
</tr>
<tr>
<td>WillBuy</td>
<td></td>
</tr>
<tr>
<td>AnyOrder</td>
<td></td>
</tr>
<tr>
<td>MktOnly</td>
<td>Market orders only</td>
</tr>
<tr>
<td>ManualAsk</td>
<td>This indicates that a market participants Ask is in a manual (NASDAQ) mode. While in this mode, automated execution is not eligible on the Ask side and can be traded through pursuant to Regulation NMS requirements.</td>
</tr>
</tbody>
</table>
6.123.3.24 virtual const char* Wombat::MamdaQuoteRecap::getQuoteQualNative () const [pure virtual]

Get the native feed quote qualifier.

**Returns:**

Native quote qualifier (a.k.a. "quote condition"). Feed-specific quote qualifier code(s). This field is provided primarily for completeness and/or troubleshooting.

**See also:**

getQuoteQual.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.25 virtual char Wombat::MamdaQuoteRecap::getShortSaleBidTick () const [pure virtual]

NASDAQ Bid Tick Indicator for Short Sale Rule Compliance.

National Bid Tick Indicator based on changes to the bid price of the National Best Bid or Offer (National BBO).

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>Up Tick. The current Best Bid Price price is higher than the previous Best Bid Price for the given NNM security.</td>
</tr>
<tr>
<td>D</td>
<td>Down Tick. The current Best Bid Price price is lower than the previous Best Bid Price for the given NNM security.</td>
</tr>
<tr>
<td>N</td>
<td>No Tick. The NASD Short Sale Rule does not apply to issue (i.e. NASDAQ SmallCap listed security).</td>
</tr>
<tr>
<td>Z</td>
<td>Unset - default value within the API</td>
</tr>
</tbody>
</table>

**Returns:**

The tick bid indicator.

Implemented in Wombat::MamdaQuoteListener.
6.123.3.26 virtual const MamaDateTime& Wombat::MamdaQuoteRecap::getAskTime () const [pure virtual]

Get the quote ask time.

Returns:

The quote ask time.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.27 virtual const MamaDateTime& Wombat::MamdaQuoteRecap::getBidTime () const [pure virtual]

Get the quote bid time.

Returns:

The quote bid time.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.28 virtual const MamaDateTime& Wombat::MamdaQuoteRecap::getQuoteDate () const [pure virtual]

Get the quote date.

Returns:

The quote date.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.29 virtual const char* Wombat::MamdaQuoteRecap::getAskIndicator () const [pure virtual]

Get the quote ask indicator.

Returns:

The quote ask indicator.

Implemented in Wombat::MamdaQuoteListener.
6.123.3.30 virtual const char* Wombat::MamdaQuoteRecap::getBidIndicator () const [pure virtual]

Get the quote bid indicator.

Returns:
The quote bid indicator.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.31 virtual mama_u32_t Wombat::MamdaQuoteRecap::getAskUpdateCount () const [pure virtual]

Get the quote ask update count.

Returns:
The quote ask update count.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.32 virtual mama_u32_t Wombat::MamdaQuoteRecap::getBidUpdateCount () const [pure virtual]

Get the quote bid update count.

Returns:
The quote bid update count.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.33 virtual double Wombat::MamdaQuoteRecap::getAskYield () const [pure virtual]

Get the quote ask yield.

Returns:
The quote ask yield.

Implemented in Wombat::MamdaQuoteListener.
virtual double Wombat::MamdaQuoteRecap::getBidYield () const
[pure virtual]

Get the quote bid yield.

Returns:
   The quote bid yield.

Implemented in Wombat::MamdaQuoteListener.

virtual const char ∗ Wombat::MamdaQuoteRecap::getAskSizesList () const  [pure virtual]

Get the ask sizes list.

Returns:
   The ask sizes list.

Implemented in Wombat::MamdaQuoteListener.

virtual const char ∗ Wombat::MamdaQuoteRecap::getBidSizesList () const  [pure virtual]

Get the bid sizes list.

Returns:
   The bid sizes list.

Implemented in Wombat::MamdaQuoteListener.

virtual char Wombat::MamdaQuoteRecap::getShortSaleCircuitBreaker () const  [pure virtual]

get the ShortSaleCircuitBreaker

Returns:
   ShortSaleCircuitBreaker
   
   • return values:
   
   • Blank: Short Sale Restriction Not in Effect.
6.123 Wombat::MamdaQuoteRecap Class Reference

• A: Short Sale Restriction Activated.
• C: Short Sale Restriction Continued.
• D: Sale Restriction Deactivated.
• E: Sale Restriction in Effect.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.38 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getBidPriceFieldState () const [pure virtual]

Get the quote bid price fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.39 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getBidSizeFieldState () const [pure virtual]

Get the quote bid size fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.40 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getBidDepthFieldState () const [pure virtual]

Get the quote bid depth fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
6.123.3.41 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getBidPartIdFieldState () const [pure virtual]

Get the quote participant identifier fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.42 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getAskPriceFieldState () const [pure virtual]

Get the quote ask price fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.43 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getAskSizeFieldState () const [pure virtual]

Get the quote ask size fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.44 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getAskDepthFieldState () const [pure virtual]

Get the quote ask depth fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
6.123.3.45 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getAskPartIdFieldState () const [pure virtual]

Get the quote ask participant identifier fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.46 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getQuoteMidPriceFieldState () const [pure virtual]

Get the quote mid price fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.47 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getQuoteQualStrFieldState () const [pure virtual]

Get the quote qualifier fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.48 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getQuoteQualNativeFieldState () const [pure virtual]

Get the native feed quote qualifier fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
6.123.3.49 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getShortSaleBidTickFieldState () const [pure virtual]

Get the short sale bid tick fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.50 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getAskTimeFieldState () const [pure virtual]

Get the quote ask time fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.51 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getBidTimeFieldState () const [pure virtual]

Get the quote bid time fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.52 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getQuoteDateFieldState () const [pure virtual]

Get the quote date fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
Get the quote ask indicator fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaQuoteListener`.

Get the quote bid indicator fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaQuoteListener`.

Get the quote ask update count fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaQuoteListener`.

Get the quote bid update count fieldState.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaQuoteListener`. 
6.123.3.57 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getAskYieldFieldState () const [pure virtual]

Get the quote ask yield fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.58 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getBidYieldFieldState () const [pure virtual]

Get the quote bid yield fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.59 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getAskSizesListFieldState () const [pure virtual]

Get the ask sizes list fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.123.3.60 virtual MamdaFieldState Wombat::MamdaQuoteRecap::getBidSizesListFieldState () const [pure virtual]

Get the bid sizes list fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
virtual MamdaFieldState Wombat::MamdaQuoteRecap::getShortSaleCircuitBreakerFieldState () const [pure virtual]

Returns:

The ShortSaleCircuitBreaker Field State. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

The documentation for this class was generated from the following file:

- MamdaQuoteRecap.h
6.124 MamdaQuoteToBookListener Class Reference

#include <MamdaQuoteToBookListener.h>

Inheritance diagram for MamdaQuoteToBookListener::

```
Wombat::MamdaMsgListener
Wombat::MamdaOrderBookListener
MamdaQuoteToBookListener
```

Public Member Functions

- **MamdaQuoteToBookListener (MamdaOrderBook *fullBook=NULL)**
  
  Create an order book listener using an optional user-provided object for the full order book.

- **virtual ~MamdaQuoteToBookListener ()**
- **virtual void addHandler (MamdaOrderBookHandler *handler)**
  
  Add a specialized order book handler.

- **virtual void setProcessEntries (bool process)**
  
  Set whether we are interested in "entry level" information at all.

- **virtual void addIgnoreEntryId (const char *id)**
  
  Add an entry ID to ignore.

- **virtual void removeIgnoreEntryId (const char *id)**
  
  Remove an entry ID to ignore.

- **virtual void setUseEntryManager (bool useManager)**
  
  Set whether to use an "entry manager" for finding entries in a book.

- **virtual void setEntryIdsAreUnique (bool uniqueIds)**
  
  Set whether the entry IDs are unique across the entire book.

- **virtual void setKeepBasicDeltas (bool keep)**
  
  Set whether to gather individual basic deltas when processing a complex delta.
- virtual void setConflateDeltas (bool conflate)
  Whether to conflate the order book deltas.

- virtual bool getConflateDeltas ()
  Whether or not order book deltas are conflated.

- virtual void setConflationInterval (double interval)
  The conflation interval.

- virtual void forceInvokeDeltaHandlers ()
  Invoke delta handlers immediately if there is a conflated delta pending.

- virtual void clearConflatedDeltas ()
  Clear any pending conflated deltas.

- virtual void setProcessProperties (bool value)
- virtual bool getProcessProperties () const

- virtual MamdaFieldState getSymbolFieldState () const
- virtual MamdaFieldState getPartIdFieldState () const
- virtual MamdaFieldState getEventSeqNumFieldState () const
- virtual MamdaFieldState getEventTimeFieldState () const
- virtual MamdaFieldState getSrcTimeFieldState () const
- virtual MamdaFieldState getActivityTimeFieldState () const
- virtual MamdaFieldState getTimeLineFieldState () const
- virtual MamdaFieldState getSendTimeFieldState () const
- virtual MamdaFieldState getMsgQualFieldState () const

- virtual void removeHandlers ()
  Remove the reference of handlers from the internal list.

- virtual void clear ()
  Clear all cached data fields.

- virtual mama_seqnum_t getBeginGapSeqNum () const
- virtual mama_seqnum_t getEndGapSeqNum () const

- virtual const MamaOrderBook * getOrderBook () const
- virtual const char * getSymbol () const
- virtual const char * getPartId () const

- virtual mama_seqnum_t getEventSeqNum () const

- virtual const MamaDateTime & getEventTime () const
- virtual const MamaDateTime & getSrcTime () const
- virtual const MamaDateTime & getActivityTime () const
- virtual const MamaDateTime & getTimeLine () const
- virtual const MamaDateTime & getSendTime () const
• virtual const MamaMsgQual & getMsgQual () const
• virtual mama_u32_t getQuoteCount () const
• virtual void setUpdateInconsistentBook (bool update)
  Whether to handle or ignore updates sent for an inconsistent book.

• virtual void setUpdateStaleBook (bool update)
  Whether to handle or ignore updates sent for a stale order book.

• virtual void setClearStaleBook (bool clear)
  Whether to clear the order book upon detection of a stale book condition.

• virtual void resolvePossiblyDuplicate (bool tf)
• virtual void setQuality (MamdaSubscription *sub, mamaQuality quality)
  Change the order book quality and invoke appropriate callbacks.

• void setQuoteSizeMultiplier (int multiplier)
• virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)
  Implementation of MamdaListener interface.

• virtual void acquireReadLock ()
  Acquire a read lock on the full book, so that an alternative running thread can view
  the book without it being changed from within the MamdaBookListener.

• virtual void releaseReadLock ()
  Release the read lock on the full book.

• virtual void acquireWriteLock ()
• virtual void releaseWriteLock ()
• virtual void getBookSnapShot (MamdaOrderBook &result)
  A thread safe function that can be called to populate a snapshot of the full book for
  the given symbol providing there already exists a MamdaBookListener for the symbol
  within the process.

### 6.124.1 Constructor & Destructor Documentation

#### 6.124.1.1 MamdaQuoteToBookListener::MamdaQuoteToBookListener

(MamdaOrderBook *fullBook = NULL)

Create an order book listener using an optional user-provided object for the full order
book.
If "fullBook" is NULL, an object will be allocated internally. If this listener is destroyed then the full order book object will only be destroyed if it was created by the listener (i.e., if fullBook was passed as NULL in this constructor).

Parameters:

- **fullBook** The MamdaOrderBook used to maintain the full book.

### 6.124.1.2 virtual MamdaQuoteToBookListener::~MamdaQuoteToBookListener () [virtual]

### 6.124.2 Member Function Documentation

#### 6.124.2.1 virtual void MamdaQuoteToBookListener::addHandler (MamdaOrderBookHandler ∗ handler) [virtual]

Add a specialized order book handler.

Currently, only one handler can (and must) be registered.

Parameters:

- **handler** The handler registered to receive order book update callbacks.

Reimplemented from Wombat::MamdaOrderBookListener.

#### 6.124.2.2 virtual void MamdaQuoteToBookListener::setProcessEntries (bool process) [virtual]

Set whether we are interested in "entry level" information at all.

Many applications only care about price level information. (Default is to process entry level information.)

Parameters:

- **process** Whether to process entries in books.

Reimplemented from Wombat::MamdaOrderBookListener.

#### 6.124.2.3 virtual void MamdaQuoteToBookListener::addIgnoreEntryId (const char ∗ id) [virtual]

Add an entry ID to ignore.
This only makes sense when the entry type is participant ID (as opposed to order ID). All order book updates for this entry ID will be ignored.

**Parameters:**

`id` The id of the participant to ignore when process book updates.

Reimplemented from `Wombat::MamdaOrderBookListener`.

### 6.124.2.4 virtual void MamdaQuoteToBookListener::removeIgnoreEntryId (const char ∗ id) [virtual]

Remove an entry ID to ignore.

**Parameters:**

`id` The id of the participant whose update will be subsequently processed as part of the book.

See also:

`addIgnoreEntryId(const char+)`

Reimplemented from `Wombat::MamdaOrderBookListener`.

### 6.124.2.5 virtual void MamdaQuoteToBookListener::setUseEntryManager (bool useManager) [virtual]

Set whether to use an "entry manager" for finding entries in a book.

Reimplemented from `Wombat::MamdaOrderBookListener`.

### 6.124.2.6 virtual void MamdaQuoteToBookListener::setEntryIdsAreUnique (bool uniqueIds) [virtual]

Set whether the entry IDs are unique across the entire book.

This is only relevant if using an "entry manager".

Reimplemented from `Wombat::MamdaOrderBookListener`.

### 6.124.2.7 virtual void MamdaQuoteToBookListener::setKeepBasicDeltas (bool keep) [virtual]

Set whether to gather individual basic deltas when processing a complex delta.
Many applications don’t need the basic deltas and will iterate over part or all of the full book (with the deltas already applied).

Reimplemented from `Wombat::MamdaOrderBookListener`.

### 6.124.2.8 virtual void MamdaQuoteToBookListener::setConflateDeltas (bool conflate) [virtual]

Whether to conflate the order book deltas.

**Parameters:**

- `conflate` Whether to conflate the order book deltas

Reimplemented from `Wombat::MamdaOrderBookListener`.

### 6.124.2.9 virtual bool MamdaQuoteToBookListener::getConflateDeltas () [virtual]

Whether or not order book deltas are conflated.

**Returns:**

- true if conflation is set to true

Reimplemented from `Wombat::MamdaOrderBookListener`.

### 6.124.2.10 virtual void MamdaQuoteToBookListener::setConflationInterval (double interval) [virtual]

The conflation interval.

**Parameters:**

- `interval` The conflation interval in seconds. Default is 0.5 seconds.

Reimplemented from `Wombat::MamdaOrderBookListener`.

### 6.124.2.11 virtual void MamdaQuoteToBookListener::forceInvokeDelta-Handlers () [virtual]

Invoke delta handlers immediately if there is a conflated delta pending.

Reimplemented from `Wombat::MamdaOrderBookListener`.
6.124.2.12 virtual void MamdaQuoteToBookListener::clearConflatedDeltas ()
        [virtual]

Clear any pending conflated deltas.
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.13 virtual void MamdaQuoteToBookListener::setProcessProperties (bool value) [virtual]

6.124.2.14 virtual bool MamdaQuoteToBookListener::getProcessProperties () const [virtual]

6.124.2.15 virtual MamdaFieldState MamdaQuoteToBookListener::getSymbolFieldState () const [virtual]

Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.16 virtual MamdaFieldState MamdaQuoteToBookListener::getPartIdFieldState () const [virtual]

Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.17 virtual MamdaFieldState MamdaQuoteToBookListener::getEventSeqNumFieldState () const [virtual]

Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.18 virtual MamdaFieldState MamdaQuoteToBookListener::getEventTimeFieldState () const [virtual]

Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.19 virtual MamdaFieldState MamdaQuoteToBookListener::getSrcTimeFieldState () const [virtual]

Reimplemented from Wombat::MamdaOrderBookListener.
6.124.2.20 virtual MamdaFieldState MamdaQuoteToBookListener::getActivityTimeFieldState () const
 [virtual]

Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.21 virtual MamdaFieldState MamdaQuoteToBookListener::getLineTimeFieldState () const
 [virtual]

Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.22 virtual MamdaFieldState MamdaQuoteToBookListener::getSendTimeFieldState () const
 [virtual]

Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.23 virtual MamdaFieldState MamdaQuoteToBookListener::getMsgQualFieldState () const
 [virtual]

Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.24 virtual void MamdaQuoteToBookListener::removeHandlers ()
 [virtual]

Remove the reference of handlers from the internal list.
Memory is not freed.
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.25 virtual void MamdaQuoteToBookListener::clear () [virtual]

Clear all cached data fields.
Reimplemented from Wombat::MamdaOrderBookListener.
6.124.2.26 virtual mama_seqnum_t MamdaQuoteToBookListener::getBeginGapSeqNum () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.27 virtual mama_seqnum_t MamdaQuoteToBookListener::getEndGapSeqNum () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.28 virtual const MamdaOrderBook* MamdaQuoteToBookListener::getOrderBook () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.29 virtual const char* MamdaQuoteToBookListener::getSymbol () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.30 virtual const char* MamdaQuoteToBookListener::getPartId () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.31 virtual mama_seqnum_t MamdaQuoteToBookListener::getEventSeqNum () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.32 virtual const MamaDateTime& MamdaQuoteToBookListener::getEventTime () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.33 virtual const MamaDateTime& MamdaQuoteToBookListener::getSrcTime () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.
6.124.2.34 virtual const MamaDateTime& MamdaQuoteToBookListener::getActivityTime () const
[virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.35 virtual const MamaDateTime& MamdaQuoteToBookListener::getLineTime () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.36 virtual const MamaDateTime& MamdaQuoteToBookListener::getSendTime () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.37 virtual const MamaMsgQual& MamdaQuoteToBookListener::getMsgQual () const [virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.38 virtual mama_u32_t MamdaQuoteToBookListener::getQuoteCount () const [virtual]

6.124.2.39 virtual void MamdaQuoteToBookListener::setUpdateInconsistentBook (bool update) [virtual]
Whether to handle or ignore updates sent for an inconsistent book.
A book may be in an inconsistent state if there has been a gap on the sequence of update (delta) messages. Default is to not update (and wait for a recap).

Parameters:

update Whether to apply updates to an inconsistent book.
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.40 virtual void MamdaQuoteToBookListener::setUpdateStaleBook (bool update) [virtual]
Whether to handle or ignore updates sent for a stale order book.
A book is in a stale state if its mamaQuality is not MAMA_QUALITY_OK.
Parameters:

update  Whether to apply updates to a stale book.

Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.41  virtual void MamdaQuoteToBookListener::setClearStaleBook (bool clear) [virtual]

Whether to clear the order book upon detection of a stale book condition.
If this is set to true, then setUpdateStaleBook() should probably have been set to false.
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.42  virtual void MamdaQuoteToBookListener::resolvePossiblyDuplicate (bool tf) [virtual]

6.124.2.43  virtual void MamdaQuoteToBookListener::setQuality (MamdaSubscription * sub, mamaQuality quality) [virtual]

Change the order book quality and invoke appropriate callbacks.
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.44  void MamdaQuoteToBookListener::setQuoteSizeMultiplier (int multiplier)

6.124.2.45  virtual void MamdaQuoteToBookListener::onMsg (MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [virtual]

Implementation of MamdaListener interface.
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.46  virtual void MamdaQuoteToBookListener::acquireReadLock () [virtual]

Acquire a read lock on the full book, so that an alternative running thread can view the book without it being changed from within the MamdaBookListener.
releaseReadLock should be called to release the lock.
Reimplemented from Wombat::MamdaOrderBookListener.
6.124.2.47 virtual void MamdaQuoteToBookListener::releaseReadLock ()
[virtual]

Release the read lock on the full book.
A companion to the acquireReadLock method.
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.48 virtual void MamdaQuoteToBookListener::acquireWriteLock ()
[virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.49 virtual void MamdaQuoteToBookListener::releaseWriteLock ()
[virtual]
Reimplemented from Wombat::MamdaOrderBookListener.

6.124.2.50 virtual void MamdaQuoteToBookListener::getBookSnapShot
(MamdaOrderBook & result) [virtual]

A thread safe function that can be called to populate a snapshot of the full book for
the given symbol providing there already exists a MamdaBookListener for the symbol
within the process.
This routine will throw a MamdaOrderBookException in the following three circum-
cstances:

• If the Mamda API hasn’t been built with multi-threading enabled, or
• If there is no listener within the process for the given symbol, or
• The listener hasn’t as yet received an initial image.

Exceptions:

<\em>MamdaOrderBookException</em> See above for details.

Reimplemented from Wombat::MamdaOrderBookListener.
The documentation for this class was generated from the following file:

• MamdaQuoteToBookListener.h
# MamdaQuoteUpdate Class Reference

MamdaQuoteUpdate is an interface that provides access to fields related to quote updates.

```cpp
#include <MamdaQuoteUpdate.h>
```

Inheritance diagram for Wombat::MamdaQuoteUpdate:

```
Wombat::MamdaQuoteUpdate
Wombat::MamdaBasicEvent
Wombat::MamdaQuoteListener
```

## Public Member Functions

- **virtual const MamaPrice & getBidPrice () const =0**
  
  Get the quote bid price.

- **virtual mama_quantity_t getBidSize () const =0**
  
  Get the quote bid size.

- **virtual mama_quantity_t getBidDepth () const =0**
  
  Get the quote bid depth.

- **virtual const char * getBidPartId () const =0**
  
  Get the quote participant identifier.

- **virtual const MamaPrice & getAskPrice () const =0**
  
  Get the quote ask price.

- **virtual mama_quantity_t getAskSize () const =0**
  
  Get the quote ask size.

- **virtual mama_quantity_t getAskDepth () const =0**
  
  Get the quote ask depth.

- **virtual const char * getAskPartId () const =0**
  
  Get the quote ask participant identifier.
• virtual const MamaPrice & getQuoteMidPrice () const =0
  Get the quote mid price.

• virtual const char * getQuoteQualStr () const =0
  Get the quote qualifier.

• virtual const char * getQuoteQualNative () const =0
  Get the native feed quote qualifier.

• virtual char getShortSaleBidTick () const =0
  NASDAQ Bid Tick Indicator for Short Sale Rule Compliance.

• virtual const MamaDateTime & getAskTime () const =0
  Get the quote ask time.

• virtual const MamaDateTime & getBidTime () const =0
  Get the quote bid time.

• virtual const MamaDateTime & getQuoteDate () const =0
  Get the quote date.

• virtual const char * getAskIndicator () const =0
  Get the quote ask indicator.

• virtual const char * getBidIndicator () const =0
  Get the quote bid indicator.

• virtual mama_u32_t getAskUpdateCount () const =0
  Get the quote ask update count.

• virtual mama_u32_t getBidUpdateCount () const =0
  Get the quote bid update count.

• virtual double getAskYield () const =0
  Get the quote ask yield.

• virtual double getBidYield () const =0
  Get the quote bid yield.

• virtual const char * getAskSizesList () const =0
  Get the ask sizes list.
• virtual const char * getBidSizesList () const =0
  Get the bid sizes list.

• virtual char getShortSaleCircuitBreaker () const =0
  get the ShortSaleCircuitBreaker

• virtual MamdaFieldState getBidPriceFieldState () const =0
  Get the quote bid price fieldState.

• virtual MamdaFieldState getBidSizeFieldState () const =0
  Get the quote bid size fieldState.

• virtual MamdaFieldState getBidDepthFieldState () const =0
  Get the quote bid depth fieldState.

• virtual MamdaFieldState getBidPartIdFieldState () const =0
  Get the quote participant identifier fieldState.

• virtual MamdaFieldState getAskPriceFieldState () const =0
  Get the quote ask price fieldState.

• virtual MamdaFieldState getAskSizeFieldState () const =0
  Get the quote ask size fieldState.

• virtual MamdaFieldState getAskDepthFieldState () const =0
  Get the quote ask depth fieldState.

• virtual MamdaFieldState getAskPartIdFieldState () const =0
  Get the quote ask participant identifier fieldState.

• virtual MamdaFieldState getQuoteMidPriceFieldState () const =0
  Get the quote mid price fieldState.

• virtual MamdaFieldState getQuoteQualStrFieldState () const =0
  Get the quote qualifier fieldState.

• virtual MamdaFieldState getQuoteQualNativeFieldState () const =0
  Get the native feed quote qualifier fieldState.

• virtual MamdaFieldState getShortSaleBidTickFieldState () const =0
  Get the short sale bid tick fieldState.
• virtual MamdaFieldState getAskTimeFieldState () const =0
  Get the quote ask time fieldState.

• virtual MamdaFieldState getBidTimeFieldState () const =0
  Get the quote bid time fieldState.

• virtual MamdaFieldState getQuoteDateFieldState () const =0
  Get the quote date fieldState.

• virtual MamdaFieldState getAskIndicatorFieldState () const =0
  Get the quote ask indicator fieldState.

• virtual MamdaFieldState getBidIndicatorFieldState () const =0
  Get the quote bid indicator fieldState.

• virtual MamdaFieldState getAskUpdateCountFieldState () const =0
  Get the quote ask update count fieldState.

• virtual MamdaFieldState getBidUpdateCountFieldState () const =0
  Get the quote bid update count fieldState.

• virtual MamdaFieldState getAskYieldFieldState () const =0
  Get the quote ask yield fieldState.

• virtual MamdaFieldState getBidYieldFieldState () const =0
  Get the quote bid yield fieldState.

• virtual MamdaFieldState getAskSizesListFieldState () const =0
  Get the ask sizes list fieldState.

• virtual MamdaFieldState getBidSizesListFieldState () const =0
  Get the bid sizes list fieldState.

• virtual MamdaFieldState getShortSaleCircuitBreakerFieldState () const =0

• virtual ~MamdaQuoteUpdate ()

6.125.1 Detailed Description

MamdaQuoteUpdate is an interface that provides access to fields related to quote updates.
6.125.2 Constructor & Destructor Documentation

6.125.2.1 virtual Wombat::MamdaQuoteUpdate::~MamdaQuoteUpdate ()
    [virtual]

6.125.3 Member Function Documentation

6.125.3.1 virtual const MamaPrice& Wombat::MamdaQuoteUpdate::getBidPrice () const
    [pure virtual]

Get the quote bid price.

Returns:

   Bid price. The highest price that the representative party/group is willing to pay to
   buy the security. For most feeds, this size is represented in round lots.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.2 virtual mama_quantity_t Wombat::MamdaQuoteUpdate::getBidSize () const
    [pure virtual]

Get the quote bid size.

Returns:

   Total share size available for the current bid price. Note: many feeds provide this
   size in terms of round lots.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.3 virtual mama_quantity_t Wombat::MamdaQuoteUpdate::getBidDepth () const
    [pure virtual]

Get the quote bid depth.

Returns:

   The total size available at the current best bid price. Only supported by some
   exchanges. The total size may differ from the "best" size in that the total may be
   aggregated from multiple sources.

Implemented in Wombat::MamdaQuoteListener.
6.125.3.4 virtual const char* Wombat::MamdaQuoteUpdate::getBidPartId () const [pure virtual]

Get the quote participant identifier.

Returns:

The identifier of the market participant (e.g. exchange or market maker) contributing the bid price field.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.5 virtual const MamaPrice& Wombat::MamdaQuoteUpdate::getAskPrice () const [pure virtual]

Get the quote ask price.

Returns:

Ask price. The lowest price that the representative party/group is willing to take to sell the security. For most feeds, this size is represented in round lots.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.6 virtual mama_quantity_t Wombat::MamdaQuoteUpdate::getAskSize () const [pure virtual]

Get the quote ask size.

Returns:

Total share size available for the current ask price. Note: many feeds provide this size in terms of round lots.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.7 virtual mama_quantity_t Wombat::MamdaQuoteUpdate::getAskDepth () const [pure virtual]

Get the quote ask depth.
Returns:

The total size available at the current best ask price. Only supported by some exchanges. The total size may differ from the "best" size in that the total may be aggregated from multiple sources.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.8 virtual const char ∗ Wombat::MamdaQuoteUpdate::getAskPartId () const [pure virtual]

Get the quote ask participant identifier.

Returns:

The identifier of the market participant (e.g. exchange or market maker) contributing the ask price field.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.9 virtual const MamaPrice& Wombat::MamdaQuoteUpdate::getQuoteMidPrice () const [pure virtual]

Get the quote mid price.

Returns:

The mid price of the current quote. Usually, this is the average of the bid and ask prices, but some exchanges provide this field explicitly (e.g. LSE).

Implemented in Wombat::MamdaQuoteListener.

6.125.3.10 virtual const char ∗ Wombat::MamdaQuoteUpdate::getQuoteQualStr () const [pure virtual]

Get the quote qualifier.

Returns:

A normalized set of qualifiers for the last quote for the security. This field may contain multiple string values, separated by the colon(;) character.
See also:

MamdaQuoteRecap::getQuoteQualStr()

Implemented in Wombat::MamdaQuoteListener.

6.125.3.11 virtual const char ∗ Wombat::MamdaQuoteUpdate::getQuoteQualNative () const [pure virtual]

Get the native feed quote qualifier.

Returns:

Native quote qualifier (a.k.a. "quote condition"). Feed-specific quote qualifier code(s). This field is provided primarily for completeness and/or troubleshooting.

See also:

getQuoteQual.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.12 virtual char Wombat::MamdaQuoteUpdate::getShortSaleBidTick () const [pure virtual]

NASDAQ Bid Tick Indicator for Short Sale Rule Compliance.

National Bid Tick Indicator based on changes to the bid price of the National Best Bid or Offer (National BBO).

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>Up Tick. The current Best Bid Price price is higher than the previous Best Bid Price for the given NNM security.</td>
</tr>
<tr>
<td>D</td>
<td>Down Tick. The current Best Bid Price price is lower than the previous Best Bid Price for the given NNM security.</td>
</tr>
<tr>
<td>N</td>
<td>No Tick. The NASD Short Sale Rule does not apply to issue (i.e. NASDAQ SmallCap listed security).</td>
</tr>
<tr>
<td>Z</td>
<td>Unset - default value within the API</td>
</tr>
</tbody>
</table>

Returns:

The tick bid indicator.

Implemented in Wombat::MamdaQuoteListener.
6.125.3.13 virtual const MamaDateTime& Wombat::MamdaQuoteUpdate::getAskTime () const [pure virtual]

Get the quote ask time.

**Returns:**

The quote ask time.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.14 virtual const MamaDateTime& Wombat::MamdaQuoteUpdate::getBidTime () const [pure virtual]

Get the quote bid time.

**Returns:**

The quote bid time.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.15 virtual const MamaDateTime& Wombat::MamdaQuoteUpdate::getQuoteDate () const [pure virtual]

Get the quote date.

**Returns:**

The quote date.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.16 virtual const char* Wombat::MamdaQuoteUpdate::getAskIndicator () const [pure virtual]

Get the quote ask indicator.

**Returns:**

The quote ask indicator.

Implemented in Wombat::MamdaQuoteListener.
virtual const char* Wombat::MamdaQuoteUpdate::getBidIndicator() const [pure virtual]

Get the quote bid indicator.

Returns:
The quote bid indicator.

Implemented in Wombat::MamdaQuoteListener.

virtual mama_u32_t Wombat::MamdaQuoteUpdate::getAskUpdateCount() const [pure virtual]

Get the quote ask update count.

Returns:
The quote ask update count.

Implemented in Wombat::MamdaQuoteListener.

virtual mama_u32_t Wombat::MamdaQuoteUpdate::getBidUpdateCount() const [pure virtual]

Get the quote bid update count.

Returns:
The quote bid update count.

Implemented in Wombat::MamdaQuoteListener.

virtual double Wombat::MamdaQuoteUpdate::getAskYield() const [pure virtual]

Get the quote ask yield.

Returns:
The quote ask yield.

Implemented in Wombat::MamdaQuoteListener.
6.125.3.21 virtual double Wombat::MamdaQuoteUpdate::getBidYield () const
[pure virtual]

Get the quote bid yield.

**Returns:**

The quote bid yield.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.22 virtual const char* Wombat::MamdaQuoteUpdate::getAskSizesList () const [pure virtual]

Get the ask sizes list.

**Returns:**

The ask sizes list.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.23 virtual const char* Wombat::MamdaQuoteUpdate::getBidSizesList () const [pure virtual]

Get the bid sizes list.

**Returns:**

The bid sizes list.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.24 virtual char Wombat::MamdaQuoteUpdate::getShortSaleCircuitBreaker () const [pure virtual]

get the ShortSaleCircuitBreaker

**Returns:**

ShortSaleCircuitBreaker

- return values:
  - Blank: Short Sale Restriction Not in Effect.
• A: Short Sale Restriction Activated.
• C: Short Sale Restriction Continued.
• D: Sale Restriction Deactivated.
• E: Sale Restriction in Effect.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.25 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getBidPriceFieldState () const [pure virtual]

Get the quote bid price fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.26 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getBidSizeFieldState () const [pure virtual]

Get the quote bid size fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.27 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getBidDepthFieldState () const [pure virtual]

Get the quote bid depth fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
6.125.3.28 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getBidPartIdFieldState () const [pure virtual]

Get the quote participant identifier fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.29 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getAskPriceFieldState () const [pure virtual]

Get the quote ask price fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.30 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getAskSizeFieldState () const [pure virtual]

Get the quote ask size fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.31 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getAskDepthFieldState () const [pure virtual]

Get the quote ask depth fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
6.125.3.32 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getAskPartIdFieldState () const [pure virtual]

Get the quote ask participant identifier fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.33 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getQuoteMidPriceFieldState () const [pure virtual]

Get the quote mid price fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.34 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getQuoteQualStrFieldState () const [pure virtual]

Get the quote qualifier fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.35 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getQuoteQualNativeFieldState () const [pure virtual]

Get the native feed quote qualifier fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
6.125.3.36 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getShortSaleBidTickFieldState () const  [pure virtual]

Get the short sale bid tick fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.37 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getAskTimeFieldState () const  [pure virtual]

Get the quote ask time fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.38 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getBidTimeFieldState () const  [pure virtual]

Get the quote bid time fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.39 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getQuoteDateFieldState () const  [pure virtual]

Get the quote date fieldState.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
6.125 Wombat::MamdaQuoteUpdate Class Reference

6.125.3.40 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getAskIndicatorFieldState () const [pure virtual]

Get the quote ask indicator fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.41 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getBidIndicatorFieldState () const [pure virtual]

Get the quote bid indicator fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.42 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getAskUpdateCountFieldState () const [pure virtual]

Get the quote ask update count fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.43 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getBidUpdateCountFieldState () const [pure virtual]

Get the quote bid update count fieldState.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
6.125.3.44 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getAskYieldFieldState () const [pure virtual]

Get the quote ask yield fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.45 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getBidYieldFieldState () const [pure virtual]

Get the quote bid yield fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.46 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getAskSizesListFieldState () const [pure virtual]

Get the ask sizes list fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

6.125.3.47 virtual MamdaFieldState Wombat::MamdaQuoteUpdate::getBidSizesListFieldState () const [pure virtual]

Get the bid sizes list fieldState.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.
virtual MamdaFieldState Wombat::MamdaQuoteUpdate::get-
ShortSaleCircuitBreakerFieldState () const [pure
virtual]

Returns:

The ShortSaleCircuitBreaker Field State. An enumeration representing field state.

Implemented in Wombat::MamdaQuoteListener.

The documentation for this class was generated from the following file:

- MamdaQuoteUpdate.h
MamdaSecStatus is an interface that provides access to the Security Status fields such as symbol announce messages.

#include <MamdaSecStatus.h>

Inheritance diagram for Wombat::MamdaSecStatus::

Public Member Functions

- virtual const char * getIssueSymbol () const =0
  The "name" of the instrument (e.g.

- virtual MamdaFieldState getIssueSymbolFieldState () const =0
  Get the issue symbol field state.

- virtual const char * getReason () const =0
  Unnormalized (feed-specific) reason associated with the current status of the security (e.g.

- virtual MamdaFieldState getReasonFieldState () const =0
  Get the reason field state.

- virtual MamdaFieldState getShortSaleCircuitBreaker () const =0
- virtual MamdaFieldState getShortSaleCircuitBreakerFieldState () const =0
- virtual char getSecurityAction () const =0
  Action related to this security.

- virtual MamdaFieldState getSecurityActionFieldState () const =0
  Get the security action field state.

- virtual const char * getSecurityType () const =0
  The security type.
6.126 Wombat::MamdaSecStatus Class Reference

- virtual MamdaFieldState getSecurityTypeFieldState () const =0
  
  Get the security type field state.

- virtual const char ∗ getSecurityStatus () const =0
  
 NYSE Technologies normalised security status.

- virtual MamdaFieldState getSecurityStatusFieldState () const =0
  
  Get the security status field state.

- virtual const char ∗ getSecurityStatusQual () const =0
  
  Security status qualifier.

- virtual MamdaFieldState getSecurityStatusQualFieldState () const =0
  
  Get the security status qualifier field state.

- virtual const char ∗ getSecurityStatusNative () const =0
  
  Original "security status" field sent by the feed.

- virtual MamdaFieldState getSecurityStatusNativeFieldState () const =0
  
  Get the native security status field state.

- virtual const char ∗ getFreeText () const =0
  
  Arbitrary free text associated with the security status change.

- virtual MamdaFieldState getFreeTextFieldState () const =0
  
  Get the free text field field state.

- virtual ~MamdaSecStatus ()

### 6.126.1 Detailed Description

MamdaSecStatus is an interface that provides access to the Security Status fields such as symbol announce messages.

### 6.126.2 Constructor & Destructor Documentation

#### 6.126.2.1 virtual Wombat::MamdaSecStatus::~MamdaSecStatus ()

[virtual]

284();
6.126.3 Member Function Documentation

6.126.3.1 virtual const char* Wombat::MamdaSecStatus::getIssueSymbol () const [pure virtual]

The "name" of the instrument (e.g. IBM, CSCO, MSFT.INCA, etc.).

Returns:
The symbol name.

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.2 virtual MamdaFieldState Wombat::MamdaSecStatus::getIssueSymbolFieldState () const [pure virtual]

Get the issue symbol field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.3 virtual const char* Wombat::MamdaSecStatus::getReason () const [pure virtual]

Unnormalized (feed-specific) reason associated with the current status of the security (e.g. halted, delayed, etc.). For normalized field, see getSecurityStatusQual.

Returns:
The reason for the current status.

See also:
getSecurityStatusQual()

Implemented in Wombat::MamdaSecStatusListener.
6.126.3.4 virtual MamdaFieldState Wombat::MamdaSecStatus::getReasonFieldState () const [pure virtual]

Get the reason field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.5 virtual char Wombat::MamdaSecStatus::getShortSaleCircuitBreaker () const [pure virtual]

See also:
MamdaSecStatusRecap::getShortSaleCircuitBreaker()

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.6 virtual MamdaFieldState Wombat::MamdaSecStatus::getShortSaleCircuitBreakerFieldState () const [pure virtual]

See also:
MamdaSecStatusRecap::getShortSaleCircuitBreakerFieldState()

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.7 virtual char Wombat::MamdaSecStatus::getSecurityAction () const [pure virtual]

Action related to this security.

- A : Add this new security.

Returns:
The security action.

Implemented in Wombat::MamdaSecStatusListener.
6.126.3.8 virtual MamdaFieldState Wombat::MamdaSecStatus::getSecurityActionFieldState () const [pure virtual]

Get the security action field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.9 virtual const char ∗ Wombat::MamdaSecStatus::getSecurityType () const [pure virtual]

The security type.

- Equity Option
- NEO Option : NEO (Index) Option
- ICS Option : ICS (Foreign Currency) Option

Returns:

The security type.

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.10 virtual MamdaFieldState Wombat::MamdaSecStatus::getSecurityTypeFieldState () const [pure virtual]

Get the security type field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.11 virtual const char ∗ Wombat::MamdaSecStatus::getSecurityStatus () const [pure virtual]

NYSE Technologies normalised security status.

See getSecurityStatusOrig for the original exchange value.
6.126 Wombat::MamdaSecStatus Class Reference

- None: No security status is known/available for this security.
- Normal: Security is open for normal quoting and trading
- Closed: Security is closed (usually before or after market open)
- Halted: Security has been halted by exchange.
- NotExist: Security does not currently exist in cache (but it might in the future).
- Deleted: Security has been deleted (e.g., merger, expiration, etc.)

**Returns:**

The normalized security status.

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.12 virtual MamdaFieldState Wombat::MamdaSecStatus::getSecurityStatusFieldState () const [pure virtual]

Get the security status field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.13 virtual const char∗ Wombat::MamdaSecStatus::getSecurityStatusQual () const [pure virtual]

Security status qualifier.

- None: No security status qualifier is known/available for this security.
- Excused: An Excused withdrawal from the market.
- Withdrawn: Non-excused withdrawal by the market maker.
- Suspended: Suspended Trading
- Resume: Resume trading/quotting after halt.
- QuoteResume: Resume quoting after halt. NASDAQ distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.
• TradeResume: Resume trading after halt. NASDAQ distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.

• ResumeTime: When the security is expected to resume trading.

• MktImbBuy: Market Imbalance - Buy. A 50,000 share or more excess of market orders to buy over market orders to sell as of 9:00am on expiration days. A MktImbBuy implies an ordinary order imbalance (OrdImbBuy).

• MktImbSell: Market Imbalance - Sell. A 50,000 share or more excess of market orders to sell over market orders to buy as of 9:00am on expiration days. A MktImbSell implies an ordinary order imbalance (OrdImbSell).

• NoMktImb: No Market Imbalance. Indicates that the imbalance of market orders for a security is less than 50,000 shares as of 9:00am on expiration days.

• MocImbBuy: Market On Close Imbalance - Buy. An excess of 50,000 share or more of MOC orders to buy over MOC orders to sell (including MOC sell plus and MOC sell short orders). A MocImbBuy implies an ordinary order imbalance (OrdImbBuy).

• MocImbSell: Market On Close Imbalance - Sell. An excess of 50,000 share or more of MOC orders to sell (not including MOC sell short and MOC sell plus orders) over orders to buy (including MOC orders to buy minus). A MocImbSell implies an ordinary order imbalance (OrdImbSell).

• NoMocImb: No Market On Close (MOC) Imbalance. The difference between the number of shares to buy MOC and the number of shares to sell MOC is less than 50,000.

• OrderImb: Non-regulatory condition: a significant imbalance of buy or sell orders exists for this security.

• OrderInf: Non-regulatory condition where there is a significant influx of orders.

• OrderImbBuy: Non-regulatory condition: a significant imbalance of buy orders exists for this security.

• OrderImbSell: Non-regulatory condition: a significant imbalance of sell orders exists for this security.

• OrderImbNone: The earlier imbalance of buy or sell orders no longer exists for this security. It also might mean that there is no imbalance to begin with.

• RangeInd: Trading Range Indication. Not an Opening Delay or Trading Halted condition: this condition is used prior to the opening of a security to denote a probable trading range (bid and offer prices, no sizes).

• ItsPreOpen: ITS pre-opening indication.
• Reserved : Reserved (e.g., CME).
• Frozen : Frozen (e.g., CME).
• PreOpen : Preopening state (e.g., CME).
• AddInfo : Additional Information. For a security that is Opening Delayed or Trading Halted, if inadequate information is disclosed during a "news dissemination or news pending" Opening Delay or Trading Halt, the Opening Delay or Trading Halt reason could be subsequently reported as "Additional Information."
• OpenDelay : Security’s opening has been delayed by exchange. This value is usually followed by another value specifying the reason for the opening delay.
• NoOpenNoResume : Indicates that trading halt or opening delay will be in effect for the remainder of the trading day.
• PriceInd : An approximation of what a security’s opening or re-opening price range (bid and offer prices, no sizes) will be when trading resumes after a delayed opening or after a trading halt.
• Equipment : Non-regulatory condition: the ability to trade this security by a participant is temporarily inhibited due to a systems, equipment or communications facility problem, or for other technical reasons.
• Filings : Not current in regulatory filings.
• News : News pending. Denotes a regulatory trading halt due to an expected news announcement which influence the security. An Opening Delay or Trading Halt may be continued once the news has been disseminated.
• NewsDissem : News Dissemination. Denotes a regulatory trading halt when relevant news influencing the security is being disseminated. Trading is suspended until the primary market determines that an adequate publication or disclosure of information has occurred.
• Listing : Listing Noncompliance.
• Operation : Operational Halt
• Info : Information Requested. Regulatory condition: more disclosure of information is requested by the exchange for this security.
• SEC : SEC Suspension.
• Times : News Resumption Times.
• Other : Other Regulatory Halt.
• Related : Related Security; In View Of Common. Non-regulatory condition: the halt or opening delay in this security is due to its relationship with another security. This condition also applies to non-common associated securities (e.g. warrants, rights, preferreds, classes, etc.) in view of the common stock.
- IPO: Upcoming IPO issue not yet trading.

**Returns:**

The normalized security status qualifier.

Implemented in `Wombat::MamdaSecStatusListener`.

6.126.3.14 virtual `MamdaFieldState` `Wombat::MamdaSecStatus::getSecurityStatusQualFieldState () const` [pure virtual]

Get the security status qualifier field state.

**Returns:**

`MamdaFieldState`. An enumeration representing field state.

Implemented in `Wombat::MamdaSecStatusListener`.

6.126.3.15 virtual const char* `Wombat::MamdaSecStatus::getSecurityStatusNative () const` [pure virtual]

Original "security status" field sent by the feed.

NYSE Technologies feed handlers also send normalized security status.

**Returns:**

The exchange provided security status.

See also:

`getSecurityStatus()`

Implemented in `Wombat::MamdaSecStatusListener`.

6.126.3.16 virtual `MamdaFieldState` `Wombat::MamdaSecStatus::getSecurityStatusNativeFieldState () const` [pure virtual]

Get the native security status field state.

**Returns:**

`MamdaFieldState`. An enumeration representing field state.
6.126 Wombat::MamdaSecStatus Class Reference

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.17 virtual const char∗ Wombat::MamdaSecStatus::getFreeText () const
[pure virtual]

Arbitrary free text associated with the security status change.

>Returns:
Free text associated with the security status change.

Implemented in Wombat::MamdaSecStatusListener.

6.126.3.18 virtual MamdaFieldState Wombat::MamdaSecStatus::getFreeTextFieldState () const
[pure virtual]

Get the free text field field state.

>Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaSecStatusListener.

The documentation for this class was generated from the following file:

• MamdaSecStatus.h
6.127 Wombat::MamdaSecStatusFields Class Reference

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing security status related fields from update messages.

```cpp
#include <MamdaSecStatusFields.h>
```

### Static Public Member Functions

- `static void setDictionary (const MamaDictionary &dictionary)`
- `static void reset ()`
  
  Reset the dictionary for security status update fields.

- `static uint16_t getMaxFid ()`
- `static bool isSet ()`

### Static Public Attributes

- `static const MamaFieldDescriptor * PUB_CLASS`
- `static const MamaFieldDescriptor * REASON`
- `static const MamaFieldDescriptor * SECURITY_ACTION`
- `static const MamaFieldDescriptor * SECURITY_TYPE`
- `static const MamaFieldDescriptor * SECURITY_STATUS`
- `static const MamaFieldDescriptor * SECURITY_STATUS_QUAL`
- `static const MamaFieldDescriptor * SECURITY_STATUS_NATIVE`
- `static const MamaFieldDescriptor * SECURITY_STATUS_TIME`
- `static const MamaFieldDescriptor * EVENT_SEQ_NUM`
- `static const MamaFieldDescriptor * FREE_TEXT`
- `static const MamaFieldDescriptor * SHORT_SALE_CIRCUIT_BREAKER`
- `static const MamaFieldDescriptor * LULDINDICATOR`
- `static const MamaFieldDescriptor * LULDTIME`

### 6.127.1 Detailed Description

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing security status related fields from update messages.

This class should be initialized prior to using the MamdaSecurityStatusListener by calling MamdaSecStatusListener.setDictionary() with a valid dictionary object which contains security status related fields.
6.127.2 Member Function Documentation

6.127.2.1 static void Wombat::MamdaSecStatusFields::setDictionary (const MamaDictionary & dictionary) [static]

6.127.2.2 static void Wombat::MamdaSecStatusFields::reset () [static]

Reset the dictionary for security status update fields.
6.127.2.3 static uint16_t Wombat::MamdaSecStatusFields::getMaxFid ()
[static]

6.127.2.4 static bool Wombat::MamdaSecStatusFields::isSet () [static]

6.127.3 Member Data Documentation

6.127.3.1 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::PUB_CLASS [static]

6.127.3.2 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::REASON [static]

6.127.3.3 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::SECURITY_ACTION [static]

6.127.3.4 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::SECURITY_TYPE [static]

6.127.3.5 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::SECURITY_STATUS [static]

6.127.3.6 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::SECURITY_STATUS_QUAL [static]

6.127.3.7 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::SECURITY_STATUS_NATIVE [static]

6.127.3.8 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::SECURITY_STATUS_TIME [static]

6.127.3.9 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::EVENT_SEQ_NUM [static]

6.127.3.10 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::FREE_TEXT [static]

6.127.3.11 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::SHORT_SALE_CIRCUIT_BREAKER [static]

6.127.3.12 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::LULDINDICATOR [static]

6.127.3.13 const MamaFieldDescriptor* Wombat::MamdaSecStatusFields::LULDTIME [static]

The documentation for this class was generated from the following file:
• MamdaSecStatusFields.h
Wombat::MamdaSecStatusHandler Class Reference

MamdaSecStatusHandler is an interface for applications that want to have an easy way to handle security status updates.

```cpp
#include <MamdaSecStatusHandler.h>
```

Inheritance diagram for Wombat::MamdaSecStatusHandler:

```
Wombat::MamdaSecStatusHandler
Wombat::MamdaSecStatusSymbolSourceAdapter
```

Public Member Functions

- virtual void onSecStatusRecap (MamdaSubscription *subscription, MamdaSecStatusListener &listener, const MamaMsg &msg, MamdaSecStatusRecap &recap)=0
- virtual void onSecStatusUpdate (MamdaSubscription *subscription, MamdaSecStatusListener &listener, const MamaMsg &msg)=0

*Method invoked when a security status update is received.*

- virtual ~MamdaSecStatusHandler ()

### 6.128.1 Detailed Description

MamdaSecStatusHandler is an interface for applications that want to have an easy way to handle security status updates.

The interface defines callback methods for security status events:

### 6.128.2 Constructor & Destructor Documentation

#### 6.128.2.1 virtual Wombat::MamdaSecStatusHandler::~MamdaSecStatusHandler () [virtual]

```cpp
61 {
}
```
6.128.3 Member Function Documentation

6.128.3.1 virtual void Wombat::MamdaSecStatusHandler::onSecStatusRecap
(MamdaSubscription * subscription, MamdaSecStatusListener & listener, const MamaMsg & msg, MamdaSecStatusRecap & recap)
[pure virtual]

Implemented in Wombat::MamdaSecStatusSymbolSourceAdapter.

6.128.3.2 virtual void Wombat::MamdaSecStatusHandler::onSecStatusUpdate
(MamdaSubscription * subscription, MamdaSecStatusListener & listener, const MamaMsg & msg) [pure virtual]

Method invoked when a security status update is received.

Parameters:

subscription The subscription which received the update

listener The listener which invoked this callback.

msg The MamaMsg that triggered this invocation.

Implemented in Wombat::MamdaSecStatusSymbolSourceAdapter.

The documentation for this class was generated from the following file:

• MamdaSecStatusHandler.h
MamdaSecurityStatusListener is a class that specializes in handling security status updates.

```cpp
#include <MamdaSecStatusListener.h>
```

Inheritance diagram for Wombat::MamdaSecStatusListener:

```
Wombat::MamdaBasicEvent Wombat::MamdaBasicRecap
Wombat::MamdaMsgListener Wombat::MamdaSecStatus Wombat::MamdaSecStatusRecap
Wombat::MamdaBasicEvent Wombat::MamdaSecStatusListener
```

**Public Member Functions**

- `MamdaSecStatusListener ()`
  
  Create a specialized security status listener.

- `virtual ~MamdaSecStatusListener ()`
- `void addHandler (MamdaSecStatusHandler *handler)`
  
  Add a specialized security status handler.

- `const char * getSymbol () const`
  
  Get the instruments string symbol.

- `const char * getPartId () const`
  
  Get the participant identifier.

- `const MamaDateTime & getSrcTime () const`
  
  Get the source time.

- `const MamaDateTime & getActivityTime () const`
  
  Get the activity time.

- `const MamaDateTime & getLineTime () const`
  
  Get the line time.

- `const MamaDateTime & getSendTime () const`
Get the send time.

• const MamaMsgQual & getMsgQual () const
  Get the message qualifier.

• const MamaDateTime & getEventTime () const
  Get the event time.

• mama_seqnum_t getEventSeqNum () const
  Get the event sequence number.

• const MamaDateTime & getLuldTime () const
  const char getLuldIndicator () const
  MamdaFieldState getSymbolFieldState () const
  Get the string symbol field state.

• MamdaFieldState getPartIdFieldState () const
  Get the participant identifier field state.

• MamdaFieldState getSrcTimeFieldState () const
  Get the source time field state.

• MamdaFieldState getActivityTimeFieldState () const
  Get the activity time field state.

• MamdaFieldState getLineTimeFieldState () const
  Get the line time of the update.

• MamdaFieldState getSendTimeFieldState () const
  Get the send time field state.

• MamdaFieldState getMsgQualFieldState () const
  Get the message qualifier field state.

• MamdaFieldState getEventTimeFieldState () const
  Get the event time field state.

• MamdaFieldState getEventSeqNumFieldState () const
  Get the event sequence number field state.

• MamdaFieldState getLuldIndicatorFieldState () const
• MamdaFieldState getLuldTimeFieldState () const
• const char * getIssueSymbol () const
The "name" of the instrument (e.g. 

- `const char * getReason () const`
  Unnormalized (feed-specific) reason associated with the current status of the security (e.g. 

- `char getSecurityAction () const`
  Action related to this security. 

- `char getShortSaleCircuitBreaker () const`
- `const char * getSecurityType () const`
  The security type. 

- `const char * getSecurityStatus () const`
  NYSE Technologies normalised security status. 

- `const char * getSecurityStatusQual () const`
  Security status qualifier.

- `const char * getSecurityStatusNative () const`
  Original "security status" field sent by the feed. 

- `const char * getSecurityStatusStr () const`
  NYSE Technologies normalised security status. 

- `const char * getSecurityStatusQualStr () const`
  Security status qualifier. 

- `MamdaSecurityStatus getSecurityStatusEnum () const`
  NYSE Technologies normalised security status. 

- `MamdaSecurityStatusQual getSecurityStatusQualifierEnum () const`
  Security status qualifier.

- `const char * getFreeText () const`
  Arbitrary free text associated with the security status change. 

- `MamdaFieldState getIssueSymbolFieldState () const`
  Get the issue symbol field state. 

- `MamdaFieldState getReasonFieldState () const`
  Get the reason field state.
• MamdaFieldState getSecurityActionFieldState () const
  Get the security action field state.

• MamdaFieldState getShortSaleCircuitBreakerFieldState () const
  MamdaFieldState getSecurityTypeFieldState () const
  Get the security type field state.

• MamdaFieldState getSecurityStatusFieldState () const
  Get the security status field state.

• MamdaFieldState getSecurityStatusQualFieldState () const
  Get the security status qualifier field state.

• MamdaFieldState getSecurityStatusNativeFieldState () const
  Get the native security status field state.

• MamdaFieldState getSecurityStatusStrFieldState () const
  MamdaFieldState getSecurityStatusQualStrFieldState () const
  MamdaFieldState getSecurityStatusEnumFieldState () const
  MamdaFieldState getSecurityStatusQualifierEnumFieldState () const
  MamdaFieldState getFreeTextFieldState () const
  Get the free text field field state.

• virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)
  Implementation of MamdaListener interface.

6.129.1 Detailed Description

MamdaSecurityStatusListener is a class that specializes in handling security status updates.

Developers provide their own implementation of the MamdaSecurityStatusHandler interface and will be delivered notifications for security status updates.

Note: The MamdaSecurityStatusListener class caches some field values. Among other reasons, caching of these fields makes it possible to provide complete certain callbacks, even when the publisher (e.g., feed handler) is only publishing deltas containing modified fields.

MamdaSecStatusListener should initialize the MamdaSecStatusFields class prior to receiving the first message by calling MamdaSecStatusFields::setDictionary() with a valid dictionary object which contains SecStatus related fields.
6.129 Wombat::MamdaSecStatusListener Class Reference

6.129.2 Constructor & Destructor Documentation

6.129.2.1 Wombat::MamdaSecStatusListener::MamdaSecStatusListener ()

Create a specialized security status listener.

6.129.2.2 virtual Wombat::MamdaSecStatusListener::~MamdaSecStatusListener () [virtual]

6.129.3 Member Function Documentation

6.129.3.1 void Wombat::MamdaSecStatusListener::addHandler (MamdaSecStatusHandler ∗ handler)

Add a specialized security status handler.

6.129.3.2 const char ∗ Wombat::MamdaSecStatusListener::getSymbol () const [virtual]

Get the instruments string symbol.

Returns:
Symbol. This is the "well-known" symbol for the security, including any symbol-ology mapping performed by the publisher.

Implements Wombat::MamdaBasicEvent.

6.129.3.3 const char ∗ Wombat::MamdaSecStatusListener::getPartId () const [virtual]

Get the participant identifier.

Returns:
Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).

Implements Wombat::MamdaBasicEvent.

6.129.3.4 const MamaDateTime& Wombat::MamdaSecStatusListener::getSrcTime () const [virtual]

Get the source time.

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
Returns:

Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.129.3.5 const MamaDateTime& Wombat::MamdaSecStatusListener::getSourceTime () const [virtual]

Get the source time.

Returns:

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements Wombat::MamdaBasicEvent.

6.129.3.6 const MamaDateTime& Wombat::MamdaSecStatusListener::getLineTime () const [virtual]

Get the line time.

Returns:

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

6.129.3.7 const MamaDateTime& Wombat::MamdaSecStatusListener::getSendTime () const [virtual]

Get the send time.

Returns:

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are
properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime()

Implements Wombat::MamdaBasicEvent.

6.129.3.8 const MamaMsgQual& Wombat::MamdaSecStatusListener::getMsgQual () const [virtual]

Get the message qualifier.

**Returns:**
Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.129.3.9 const MamaDateTime& Wombat::MamdaSecStatusListener::getEventTime () const [virtual]

Get the event time.

**Returns:**
Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.129.3.10 mama_seqnum_t Wombat::MamdaSecStatusListener::getEventSeqNum () const [virtual]

Get the event sequence number.

**Returns:**
Source sequence number. The exchange generated sequence number.

Implements Wombat::MamdaBasicEvent.
6.129.3.11  const MamaDateTime& Wombat::MamdaSecStatusListener::getLuldTime () const

6.129.3.12 const char Wombat::MamdaSecStatusListener::getLuldIndicator () const

6.129.3.13 MamdaFieldState Wombat::MamdaSecStatusListener::getSymbolFieldState () const  [virtual]

Get the string symbol field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.129.3.14 MamdaFieldState Wombat::MamdaSecStatusListener::getPartIdFieldState () const  [virtual]

Get the participant identifier field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.129.3.15 MamdaFieldState Wombat::MamdaSecStatusListener::getSrcTimeFieldState () const  [virtual]

Get the source time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.129.3.16 MamdaFieldState Wombat::MamdaSecStatusListener::getActivityTimeFieldState () const  [virtual]

Get the activity time field state.
Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.129.3.17 MamdaFieldState Wombat::MamdaSecStatusListener::getLineTimeFieldState () const [virtual]

Get the line time of the update.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.129.3.18 MamdaFieldState Wombat::MamdaSecStatusListener::getSendTimeFieldState () const [virtual]

Get the send time field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.129.3.19 MamdaFieldState Wombat::MamdaSecStatusListener::getMsgQualFieldState () const [virtual]

Get the message qualifier field state.

Returns:

Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.129.3.20 MamdaFieldState Wombat::MamdaSecStatusListener::getEventTimeFieldState () const [virtual]

Get the event time field state.
Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.129.3.21 MamdaFieldState Wombat::MamdaSecStatusListener::getEventSeqNumFieldState () const
[virtual]

Get the event sequence number field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.129.3.22 MamdaFieldState Wombat::MamdaSecStatusListener::getLuldIndicatorFieldState () const

6.129.3.23 MamdaFieldState Wombat::MamdaSecStatusListener::getLuldTimeFieldState () const

6.129.3.24 const char* Wombat::MamdaSecStatusListener::getIssueSymbol () const
[virtual]

The "name" of the instrument (e.g. IBM, CSCO, MSFT.INCA, etc.).

Returns:

The symbol name.

Implements Wombat::MamdaSecStatus.

6.129.3.25 const char* Wombat::MamdaSecStatusListener::getReason () const
[virtual]

Unnormalized (feed-specific) reason associated with the current status of the security (e.g. halted, delayed, etc.). For normalized field, see getSecurityStatusQual.
Returns:
   The reason for the current status.

See also:
   getSecurityStatusQual()

Implements Wombat::MamdaSecStatus.

6.129.3.26 char Wombat::MamdaSecStatusListener::getSecurityAction () const
   [virtual]

Action related to this security.

- A : Add this new security.

Returns:
   The security action.

Implements Wombat::MamdaSecStatus.

6.129.3.27 char Wombat::MamdaSecStatusListener::getShortSaleCircuitBreaker () const [virtual]

See also:
   MamdaSecStatusRecap::getShortSaleCircuitBreaker()

Implements Wombat::MamdaSecStatus.

6.129.3.28 const char* Wombat::MamdaSecStatusListener::getSecurityType () const [virtual]

The security type.

- Equity Option
- NEO Option : NEO (Index) Option
- ICS Option : ICS (Foreign Currency) Option

Returns:
   The security type.

Implements Wombat::MamdaSecStatus.

---

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.129.3.29 const char* Wombat::MamdaSecStatusListener::getSecurityStatus () const  [virtual]

NYSE Technologies normalised security status.
See getSecurityStatusOrig for the original exchange value.

- None : No security status is known/available for this security.
- Normal : Security is open for normal quoting and trading
- Closed : Security is closed (usually before or after market open)
- Halted : Security has been halted by exchange.
- NotExist : Security does not currently exist in cache (but it might in the future).
- Deleted : Security has been deleted (e.g., merger, expiration, etc.)

Returns:
The normalized security status.

Implements Wombat::MamdaSecStatus.

6.129.3.30 const char* Wombat::MamdaSecStatusListener::getSecurityStatusQual () const  [virtual]

Security status qualifier.

- None : No security status qualifier is known/available for this security.
- Excused : An Excused withdrawal from the market.
- Withdrawn : Non-excused withdrawal by the market maker.
- Suspended : Suspended Trading
- Resume : Resume trading/quotting after halt.
- QuoteResume : Resume quoting after halt. NASDAQ distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.
- TradeResume : Resume trading after halt. NASDAQ distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.
- ResumeTime : When the security is expected to resume trading.
- **MktImbBuy**: Market Imbalance - Buy. A 50,000 share or more excess of market orders to buy over market orders to sell as of 9:00am on expiration days. A MktImbBuy implies an ordinary order imbalance (OrdImbBuy).

- **MktImbSell**: Market Imbalance - Sell. A 50,000 share or more excess of market orders to sell over market orders to buy as of 9:00am on expiration days. A MktImbSell implies an ordinary order imbalance (OrdImbSell).

- **NoMktImb**: No Market Imbalance. Indicates that the imbalance of market orders for a security is less than 50,000 shares as of 9:00am on expiration days.

- **MocImbBuy**: Market On Close Imbalance - Buy. An excess of 50,000 share or more of MOC orders to buy over MOC orders to sell (including MOC sell plus and MOC sell short orders). A MocImbBuy implies an ordinary order imbalance (OrdImbBuy).

- **MocImbSell**: Market On Close Imbalance - Sell. An excess of 50,000 share or more of MOC orders to sell (not including MOC sell short and MOC sell plus orders) over orders to buy (including MOC orders to buy minus). A MocImbSell implies an ordinary order imbalance (OrdImbSell).

- **NoMocImb**: No Market On Close (MOC) Imbalance. The difference between the number of shares to buy MOC and the number of shares to sell MOC is less than 50,000.

- **OrderImb**: Non-regulatory condition: a significant imbalance of buy or sell orders exists for this security.

- **OrderInf**: Non-regulatory condition where there is a significant influx of orders.

- **OrderImbBuy**: Non-regulatory condition: a significant imbalance of buy orders exists for this security.

- **OrderImbSell**: Non-regulatory condition: a significant imbalance of sell orders exists for this security.

- **OrderImbNone**: The earlier imbalance of buy or sell orders no longer exists for this security. It also might mean that there is no imbalance to begin with.

- **RangeInd**: Trading Range Indication. Not an Opening Delay or Trading Halted condition: this condition is used prior to the opening of a security to denote a probable trading range (bid and offer prices, no sizes).

- **ItsPreOpen**: ITS pre-opening indication.

- **Reserved**: Reserved (e.g., CME).

- **Frozen**: Frozen (e.g., CME).

- **PreOpen**: Preopening state (e.g., CME).
• AddInfo : Additional Information. For a security that is Opening Delayed or Trading Halted, if inadequate information is disclosed during a "news dissemination or news pending" Opening Delay or Trading Halt, the Opening Delay or Trading Halt reason could be subsequently reported as "Additional Information."

• OpenDelay : Security’s opening has been delayed by exchange. This value is usually followed by another value specifying the reason for the opening delay.

• NoOpenNoResume : Indicates that trading halt or opening delay will be in effect for the remainder of the trading day.

• PriceInd : An approximation of what a security’s opening or re-opening price range (bid and offer prices, no sizes) will be when trading resumes after a delayed opening or after a trading halt.

• Equipment : Non-regulatory condition: the ability to trade this security by a participant is temporarily inhibited due to a systems, equipment or communications facility problem, or for other technical reasons.

• Filings : Not current in regulatory filings.

• News : News pending. Denotes a regulatory trading halt due to an expected news announcement which influence the security. An Opening Delay or Trading Halt may be continued once the news has been disseminated.

• NewsDissem : News Dissemination. Denotes a regulatory trading halt when relevant news influencing the security is being disseminated. Trading is suspended until the primary market determines that an adequate publication or disclosure of information has occurred.

• Listing : Listing Noncompliance.

• Operation : Operational Halt

• Info : Information Requested. Regulatory condition: more disclosure of information is requested by the exchange for this security.

• SEC : SEC Suspension.

• Times : News Resumption Times.

• Other : Other Regulatory Halt.

• Related : Related Security; In View Of Common. Non-regulatory condition: the halt or opening delay in this security is due to its relationship with another security. This condition also applies to non-common associated securities (e.g. warrants, rights, preferreds, classes, etc.) in view of the common stock.

• IPO : Upcoming IPO issue not yet trading.
Returns:

The normalized security status qualifier.

Implements Wombat::MamdaSecStatus.

6.129.3.31 const char ∗ Wombat::MamdaSecStatusListener::getSecurityStatusNative() const [virtual]

Original "security status" field sent by the feed.
NYSE Technologies feed handlers also send normalized security status.

Returns:

The exchange provided security status.

See also:

getSecurityStatus()

Implements Wombat::MamdaSecStatus.

6.129.3.32 const char ∗ Wombat::MamdaSecStatusListener::getSecurityStatusStr() const [virtual]

NYSE Technologies normalised security status.
See getSecurityStatusOrig for the original exchange value.

- None: No security status is known/available for this security.
- Normal: Security is open for normal quoting and trading
- Closed: Security is closed (usually before or after market open)
- Halted: Security has been halted by exchange.
- NotExist: Security does not currently exist in cache (but it might in the future).
- Deleted: Security has been deleted (e.g., merger, expiration, etc.)
- Auction: Security has done into auction.
- Crossing: Security crossing
- Unknown: Security status is currently unknown.

Returns:

The normalized security status.

Implements Wombat::MamdaSecStatusRecap.

---

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
Security status qualifier.

- None: No security status qualifier is known/available for this security.
- Excused: An Excused withdrawal from the market.
- Withdrawn: Non-excused withdrawal by the market maker.
- Suspended: Suspended Trading
- Resume: Resume trading/quotting after halt.
- QuoteResume: Resume quoting after halt. Nasdaq distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.
- TradeResume: Resume trading after halt. Nasdaq distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.
- ResumeTime: When the security is expected to resume trading.
- MktImbBuy: Market Imbalance - Buy. A 50,000 share or more excess of market orders to buy over market orders to sell as of 9:00am on expiration days. A MktImbBuy implies an ordinary order imbalance (OrdImbBuy).
- MktImbSell: Market Imbalance - Sell. A 50,000 share or more excess of market orders to sell over market orders to buy as of 9:00am on expiration days. A MktImbSell implies an ordinary order imbalance (OrdImbSell).
- NoMktImb: No Market Imbalance. Indicates that the imbalance of market orders for a security is less than 50,000 shares as of 9:00am on expiration days.
- MocImbBuy: Market On Close Imbalance - Buy. An excess of 50,000 share or more of MOC orders to buy over MOC orders to sell (including MOC sell plus and MOC sell short orders). A MocImbBuy implies an ordinary order imbalance (OrdImbBuy).
- MocImbSell: Market On Close Imbalance - Sell. An excess of 50,000 share or more of MOC orders to sell (not including MOC sell short and MOC sell plus orders) over orders to buy (including MOC orders to buy minus). A MocImbSell implies an ordinary order imbalance (OrdImbSell).
- NoMocImb: No Market On Close (MOC) Imbalance. The difference between the number of shares to buy MOC and the number of shares to sell MOC is less than 50,000.
- OrderImb : Non-regulatory condition: a significant imbalance of buy or sell orders exists for this security.

- OrderInf : Non-regulatory condition where there is a significant influx of orders.

- OrderImbBuy : Non-regulatory condition: a significant imbalance of buy orders exists for this security.

- OrderImbSell : Non-regulatory condition: a significant imbalance of sell orders exists for this security.

- OrderImbNone : The earlier imbalance of buy or sell orders no longer exists for this security. It also might mean that there is no imbalance to begin with.

- RangeInd : Trading Range Indication. Not an Opening Delay or Trading Halted condition: this condition is used prior to the opening of a security to denote a probable trading range (bid and offer prices, no sizes).

- ItsPreOpen : ITS pre-opening indication.

- Reserved : Reserved (e.g., CME).

- Frozen : Frozen (e.g., CME).

- PreOpen : Preopening state (e.g., CME).

- AddInfo : Additional Information. For a security that is Opening Delayed or Trading Halted, if inadequate information is disclosed during a "news dissemination or news pending" Opening Delay or Trading Halt, the Opening Delay or Trading Halt reason could be subsequently reported as "Additional Information."

- OpenDelay : Security’s opening has been delayed by exchange. This value is usually followed by another value specifying the reason for the opening delay.

- NoOpenNoResume : Indicates that trading halt or opening delay will be in effect for the remainder of the trading day.

- PriceInd : An approximation of what a security’s opening or re-opening price range (bid and offer prices, no sizes) will be when trading resumes after a delayed opening or after a trading halt.

- Equipment : Non-regulatory condition: the ability to trade this security by a participant is temporarily inhibited due to a systems, equipment or communications facility problem, or for other technical reasons.

- Filings : Not current in regulatory filings.

- News : News pending. Denotes a regulatory trading halt due to an expected news announcement which influence the security. An Opening Delay or Trading Halt may be continued once the news has been disseminated.
• NewsDissem : News Dissemination. Denotes a regulatory trading halt when relevant news influencing the security is being disseminated. Trading is suspended until the primary market determines that an adequate publication or disclosure of information has occurred.

• Listing : Listing Noncompliance.

• Operation : Operational Halt

• Info : Information Requested. Regulatory condition: more disclosure of information is requested by the exchange for this security.

• SEC : SEC Suspension.

• Times : News Resumption Times.

• Other : Other Regulatory Halt.

• Related : Related Security; In View Of Common. Non-regulatory condition: the halt or opening delay in this security is due to its relationship with another security. This condition also applies to non-common associated securities (e.g. warrants, rights, preferreds, classes, etc.) in view of the common stock.

• IPO : Upcoming IPO issue not yet trading.

Returns:
The normalized security status qualifier.

Implements Wombat::MamdaSecStatusRecap.

6.129.3.34 MamdaSecurityStatus Wombat::MamdaSecStatusListener::getSecurityStatusEnum () const
[virtual]

NYSE Technologies normalised security status.

See getSecurityStatusOrig for the original exchange value.

• SECURITY_STATUS_NONE : No security status is known/available for this security.

• SECURITY_STATUS_NORMAL : Security is open for normal quoting and trading

• SECURITY_STATUS_CLOSED : Security is closed (usually before or after market open)

• SECURITY_STATUS_HALTED : Security has been halted by exchange.
• SECURITY_STATUS_NOT_EXIST : Security does not currently exist in cache (but it might in the future).

• SECURITY_STATUS_DELETED : Security has been deleted (e.g., merger, expiration, etc.)

• SECURITY_STATUS_AUCTION : Security has gone into auction.

• SECURITY_STATUS_CROSSING : Security crossing

• SECURITY_STATUS_UNKNOWN : Security status is currently unknown.

**Returns:**

The normalized security status as an enumerated value.

Implements Wombat::MamdaSecStatusRecap.

### 6.129.3.35 MamdaSecurityStatusQual

**Wombat::MamdaSecStatusListener::getSecurityStatusQualifierEnum () const**

[virtual]

Security status qualifier.

• SECURITY_STATUS_QUAL_NONE : No security status qualifier is known/available for this security.

• SECURITY_STATUS_QUAL_EXCUSED : An Excused withdrawal from the market.

• SECURITY_STATUS_QUAL_WITHDRAWN : Non-excused withdrawal by the market maker.

• SECURITY_STATUS_QUAL_SUSPENDED : Suspended Trading

• SECURITY_STATUS_QUAL_RESUME : Resume trading/quotes after halt.

• SECURITY_STATUS_QUAL_QUOTE_RESUME : Resume quoting after halt.
  Nasdaq distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.

• SECURITY_STATUS_QUAL_TRADE_RESUME : Resume trading after halt.
  Nasdaq distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.

• SECURITY_STATUS_QUAL_RESUME_TIME : When the security is expected to resume trading.
• SECURITY_STATUS_QUAL_MKT_IMB_BUY : Market Imbalance - Buy. A 50,000 share or more excess of market orders to buy over market orders to sell as of 9:00am on expiration days. A MktImbBuy implies an ordinary order imbalance (OrdImbBuy).

• SECURITY_STATUS_QUAL_MKT_IMB_SELL : Market Imbalance - Sell. A 50,000 share or more excess of market orders to sell over market orders to buy as of 9:00am on expiration days. A MktImbSell implies an ordinary order imbalance (OrdImbSell).

• SECURITY_STATUS_QUAL_NO_MKT_IMB : No Market Imbalance. Indicates that the imbalance of market orders for a security is less than 50,000 shares as of 9:00am on expiration days.

• SECURITY_STATUS_QUAL_MOC_IMB_BUY : Market On Close Imbalance - Buy. An excess of 50,000 share or more of MOC orders to buy over MOC orders to sell (including MOC sell plus and MOC sell short orders). A MocImbBuy implies an ordinary order imbalance (OrdImbBuy).

• SECURITY_STATUS_QUAL_MOC_IMB_SELL : Market On Close Imbalance - Sell. An excess of 50,000 share or more of MOC orders to sell (not including MOC sell short and MOC sell plus orders) over orders to buy (including MOC orders to buy minus). A MocImbSell implies an ordinary order imbalance (OrdImbSell).

• SECURITY_STATUS_QUAL_NO_MOC_IMB : No Market On Close (MOC) Imbalance. The difference between the number of shares to buy MOC and the number of shares to sell MOC is less than 50,000.

• SECURITY_STATUS_QUAL_ORDER_IMB : Non-regulatory condition: a significant imbalance of buy orders exists for this security.

• SECURITY_STATUS_QUAL_ORDER_IMB_SELL : Non-regulatory condition: a significant imbalance of sell orders exists for this security.

• SECURITY_STATUS_QUAL_ORDER_OMB_NONE : The earlier imbalance of buy or sell orders no longer exists for this security. It also might mean that there is no imbalance to begin with.

• SECURITY_STATUS_QUAL_RANGE_IND : Trading Range Indication. Not an Opening Delay or Trading Halted condition: this condition is used prior to the opening of a security to denote a probable trading range (bid and offer prices, no sizes).

• SECURITY_STATUS_QUAL_ITS_PRE_OPEN : ITS pre-opening indication.

• SECURITY_STATUS_QUAL_RESERVED : Reserved (e.g., CME).

• SECURITY_STATUS_QUAL_FROZEN : Frozen (e.g., CME).
• SECURITY_STATUS_QUAL_PRE_OPEN : Preopening state (e.g., CME).

• SECURITY_STATUS_QUAL_ADD_INFO : Additional Information. For a security that is Opening Delayed or Trading Halted, if inadequate information is disclosed during a "news dissemination or news pending" Opening Delay or Trading Halt, the Opening Delay or Trading Halt reason could be subsequently reported as "Additional Information."

• SECURITY_STATUS_QUAL_OPEN_DELAY : Security’s opening has been delayed by exchange. This value is usually followed by another value specifying the reason for the opening delay.

• SECURITY_STATUS_QUAL_NO_OPEN_RESUME : Indicates that trading halt or opening delay will be in effect for the remainder of the trading day.

• SECURITY_STATUS_QUAL_PRICE_IND : An approximation of what a security’s opening or re-opening price range (bid and offer prices, no sizes) will be when trading resumes after a delayed opening or after a trading halt.

• SECURITY_STATUS_QUAL_EQUIPMENT : Non-regulatory condition: the ability to trade this security by a participant is temporarily inhibited due to a systems, equipment or communications facility problem, or for other technical reasons.

• SECURITY_STATUS_QUAL_FILINGS : Not current in regulatory filings.

• SECURITY_STATUS_QUAL_NEWS : News pending. Denotes a regulatory trading halt due to an expected news announcement which influence the security. An Opening Delay or Trading Halt may be continued once the news has been disseminated.

• SECURITY_STATUS_QUAL_NEWS_DISSEM : News Dissemination. Denotes a regulatory trading halt when relevant news influencing the security is being disseminated. Trading is suspended until the primary market determines that an adequate publication or disclosure of information has occurred.

• SECURITY_STATUS_QUAL_LISTING : Listing Noncompliance.

• SECURITY_STATUS_QUAL_OPERATION : Operational Halt

• SECURITY_STATUS_QUAL_INFO : Information Requested. Regulatory condition: more disclosure of information is requested by the exchange for this security.

• SECURITY_STATUS_QUAL_SEC : SEC Suspension.

• SECURITY_STATUS_QUAL_TIMES : News Resumption Times.

• SECURITY_STATUS_QUAL_OTHER : Other Regulatory Halt.
• SECURITY_STATUS_QUAL_RELATED : Related Security; In View Of Common. Non-regulatory condition: the halt or opening delay in this security is due to its relationship with another security. This condition also applies to non-common associated securities (e.g. warrants, rights, preferreds, classes, etc.) in view of the common stock.

• SECURITY_STATUS_QUAL_IPO : Upcoming IPO issue not yet trading.

**Returns:**

The normalized security status qualifier.

Implements Wombat::MamdaSecStatusRecap.

6.129.3.36 `const char* Wombat::MamdaSecStatusListener::getFreeText () const` [virtual]

Arbitrary free text associated with the security status change.

**Returns:**

Free text associated with the security status change.

Implements Wombat::MamdaSecStatus.

6.129.3.37 `MamdaFieldState Wombat::MamdaSecStatusListener::getIssueSymbolFieldState () const` [virtual]

Get the issue symbol field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaSecStatus.

6.129.3.38 `MamdaFieldState Wombat::MamdaSecStatusListener::getReasonFieldState () const` [virtual]

Get the reason field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaSecStatus.
6.129 Wombat::MamdaSecStatusListener Class Reference

6.129.3.39 **MamdaFieldState** Wombat::MamdaSecStatusListener::getSecurityActionFieldState ( ) const [virtual]

Get the security action field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaSecStatus.

6.129.3.40 **MamdaFieldState** Wombat::MamdaSecStatusListener::getShortSaleCircuitBreakerFieldState ( ) const [virtual]

See also:

MamdaSecStatusRecap::getShortSaleCircuitBreakerFieldState()

Implements Wombat::MamdaSecStatus.

6.129.3.41 **MamdaFieldState** Wombat::MamdaSecStatusListener::getSecurityTypeFieldState ( ) const [virtual]

Get the security type field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaSecStatus.

6.129.3.42 **MamdaFieldState** Wombat::MamdaSecStatusListener::getSecurityStatusFieldState ( ) const [virtual]

Get the security status field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaSecStatus.
6.129.3.43 MamdaFieldState Wombat::MamdaSecStatusListener::getSecurityStatusQualFieldState () const
[virtual]

Get the security status qualifier field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaSecStatus.

6.129.3.44 MamdaFieldState Wombat::MamdaSecStatusListener::getSecurityStatusNativeFieldState () const
[virtual]

Get the native security status field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaSecStatus.

6.129.3.45 MamdaFieldState Wombat::MamdaSecStatusListener::getSecurityStatusStrFieldState () const
[virtual]

Returns:
The normalized security status field state.

Implements Wombat::MamdaSecStatusRecap.

6.129.3.46 MamdaFieldState Wombat::MamdaSecStatusListener::getSecurityStatusQualStrFieldState () const
[virtual]

Returns:
The normalized security status qualifier field state.

Implements Wombat::MamdaSecStatusRecap.
6.129.3.47 **MamdaFieldState** Wombat::MamdaSecStatusListener::getSecurityStatusEnumFieldState () const [virtual]

**Returns:**

The normalized security status (enumerated value) field state

Implements Wombat::MamdaSecStatusRecap.

6.129.3.48 **MamdaFieldState** Wombat::MamdaSecStatusListener::getSecurityStatusQualifierEnumFieldState () const [virtual]

**Returns:**

The normalized security status qualifier field state

Implements Wombat::MamdaSecStatusRecap.

6.129.3.49 **MamdaFieldState** Wombat::MamdaSecStatusListener::getFreeTextFieldState () const [virtual]

Get the free text field field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaSecStatus.

6.129.3.50 **virtual void** Wombat::MamdaSecStatusListener::onMsg (MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [virtual]

Implementation of MamdaListener interface.

Implements Wombat::MamdaMsgListener.

The documentation for this class was generated from the following file:

- MamdaSecStatusListener.h
6.130 Wombat::MamdaSecStatusRecap Class Reference

MamdaSecStatus is an interface that provides access to the Security Status fields such as symbol announce messages.

```cpp
#include <MamdaSecStatusRecap.h>
```

Inheritance diagram for Wombat::MamdaSecStatusRecap:

```
Wombat::MamdaBasicRecap
  ↓
Wombat::MamdaSecStatusRecap
  ↓
Wombat::MamdaSecStatusListener
```

Public Member Functions

- virtual const char * `getIssueSymbol()` const =0
  
  The "name" of the instrument (e.g.

- virtual `MamdaFieldState getIssueSymbolFieldState()` const =0
  
  The symbol field state.

- virtual const char * `getReason()` const =0
  
  Unnormalized (feed-specific) reason associated with the current status of the security (e.g.

- virtual `MamdaFieldState getReasonFieldState()` const =0
- virtual char `getShortSaleCircuitBreaker()` const =0
  
  get the ShortSaleCircuitBreaker

- virtual `MamdaFieldState getShortSaleCircuitBreakerFieldState()` const =0
- virtual char `getSecurityAction()` const =0
  
  Action related to this security.

- virtual `MamdaFieldState getSecurityActionFieldState()` const =0
- virtual const char * `getSecurityType()` const =0
  
  The security type.
6.130.1 Detailed Description

MamdaSecStatus is an interface that provides access to the Security Status fields such as symbol announce messages.
6.130.2 Constructor & Destructor Documentation

6.130.2.1 virtual Wombat::MamdaSecStatusRecap::~MamdaSecStatusRecap()
   [virtual]

433 {};

6.130.3 Member Function Documentation

6.130.3.1 virtual const char∗ Wombat::MamdaSecStatusRecap::getIssueSymbol() const
   [pure virtual]

The "name" of the instrument (e.g. IBM, CSCO, MSFT.INCA, etc.).

Returns:

   The symbol name.

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.2 virtual MamdaFieldState Wombat::MamdaSecStatusRecap::getIssueSymbolFieldState() const
   [pure virtual]

The symbol field state.

Returns:

   The symbol Field State.

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.3 virtual const char∗ Wombat::MamdaSecStatusRecap::getReason() const
   [pure virtual]

Unnormalized (feed-specific) reason associated with the current status of the security
(e.g. halted, delayed, etc.). For normalized field, see getSecurityStatusQual.

Returns:

   The reason for the current status.
See also:

getSecurityStatusQual()

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.4 virtual MamdaFieldState Wombat::MamdaSecStatusRecap::getReasonFieldState () const [pure virtual]

Returns:

The reason field state

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.5 virtual char Wombat::MamdaSecStatusRecap::getShortSaleCircuitBreaker () const [pure virtual]

get the ShortSaleCircuitBreaker

Returns:

ShortSaleCircuitBreaker

• return values:
  • Blank: Short Sale Restriction Not in Effect.
  • A: Short Sale Restriction Activiated.
  • C: Short Sale Restriction Continued.
  • D: Sale Restriction Deactivated.
  • E: Sale Restriction in Effect.

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.6 virtual MamdaFieldState Wombat::MamdaSecStatusRecap::getShortSaleCircuitBreakerFieldState () const [pure virtual]

Returns:

The ShortSaleCircuitBreaker Field State.

Implemented in Wombat::MamdaSecStatusListener.
6.130.3.7 virtual char Wombat::MamdaSecStatusRecap::getSecurityAction () const [pure virtual]

Action related to this security.

- A : Add this new security.

Returns:

The security action.

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.8 virtual MamdaFieldState Wombat::MamdaSecStatusRecap::getSecurityActionFieldState () const [pure virtual]

Returns:

The security action field state

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.9 virtual const char* Wombat::MamdaSecStatusRecap::getSecurityType () const [pure virtual]

The security type.

- Equity Option
- NEO Option : NEO (Index) Option
- ICS Option : ICS (Foreign Currency) Option

Returns:

The security type.

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.10 virtual MamdaFieldState Wombat::MamdaSecStatusRecap::getSecurityTypeFieldState () const [pure virtual]

The security type field state.
Returns:

The security type field state

Implemented in `Wombat::MamdaSecStatusListener`.

### 6.130.3.11 virtual const char ∗ Wombat::MamdaSecStatusRecap::getSecurityStatus () const [pure virtual]

Deprecated.

Use `getSecurityStatusEnum()` to return the security status as an enumerated value, or `getSecurityStatusStr()` to return it as a const char ∗.

Implemented in `Wombat::MamdaSecStatusListener`.

### 6.130.3.12 virtual `MamdaSecurityStatus` Wombat::MamdaSecStatusRecap::getSecurityStatusEnum () const [pure virtual]

NYSE Technologies normalised security status.

See `getSecurityStatusOrig` for the original exchange value.

- SECURITY_STATUS_NONE : No security status is known/available for this security.
- SECURITY_STATUS_NORMAL : Security is open for normal quoting and trading
- SECURITY_STATUS_CLOSED : Security is closed (usually before or after market open)
- SECURITY_STATUS_HALTED : Security has been halted by exchange.
- SECURITY_STATUS_NOT_EXIST : Security does not currently exist in cache (but it might in the future).
- SECURITY_STATUS_DELETED : Security has been deleted (e.g., merger, expiration, etc.)
- SECURITY_STATUS_AUCTION : Security has gone into auction.
- SECURITY_STATUS_CROSSING : Security crossing
- SECURITY_STATUS_UNKNOWN : Security status is currently unknown.
Returns:

The normalized security status as an enumerated value.

Implemented in `Wombat::MamdaSecStatusListener`.

6.130.3.13 virtual `MamdaFieldState` `Wombat::MamdaSecStatusRecap::getSecurityStatusEnumFieldState()` const [pure virtual]

Returns:

The normalized security status (enumerated value) field State

Implemented in `Wombat::MamdaSecStatusListener`.

6.130.3.14 virtual const char* `Wombat::MamdaSecStatusRecap::getSecurityStatusStr()` const [pure virtual]

NYSE Technologies normalised security status.
See `getSecurityStatusOrig` for the original exchange value.

- None : No security status is known/available for this security.
- Normal : Security is open for normal quoting and trading
- Closed : Security is closed (usually before or after market open)
- Halted : Security has been halted by exchange.
- NotExist : Security does not currently exist in cache (but it might in the future).
- Deleted : Security has been deleted (e.g., merger, expiration, etc.)
- Auction : Security has done into auction.
- Crossing : Security crossing
- Unknown : Security status is currently unknown.

Returns:

The normalized security status.

Implemented in `Wombat::MamdaSecStatusListener`. 
virtual MamdaFieldState Wombat::MamdaSecStatusRecap::getSecurityStatusStrFieldState () const [pure virtual]

Returns:

The normalized security status field state.

Implemented in Wombat::MamdaSecStatusListener.

virtual const char * Wombat::MamdaSecStatusRecap::getSecurityStatusQual () const [pure virtual]

Deprecated.

Use getSecurityStatusQualifierEnum() to return the security status qualifier as an enumerated value, or getSecurityStatusQualifierStr() to return it as a const char*.

Implemented in Wombat::MamdaSecStatusListener.

virtual MamdaSecurityStatusQual Wombat::MamdaSecStatusRecap::getSecurityStatusQualifierEnum () const [pure virtual]

Security status qualifier.

- SECURITY_STATUS_QUAL_NONE : No security status qualifier is known/available for this security.
- SECURITY_STATUS_QUAL_EXCUSED : An Excused withdrawal from the market.
- SECURITY_STATUS_QUAL_WITHDRAWN : Non-excused withdrawal by the market maker.
- SECURITY_STATUS_QUAL_SUSPENDED : Suspended Trading
- SECURITY_STATUS_QUAL_RESUME : Resume trading/quoting after halt.
- SECURITY_STATUS_QUALQUOTE_RESUME : Resume quoting after halt. Nasdaq distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.
- SECURITY_STATUS_QUAL_TRADE_RESUME : Resume trading after halt. Nasdaq distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.
• SECURITY_STATUS_QUAL_RESUME_TIME : When the security is expected to resume trading.

• SECURITY_STATUS_QUAL_MKT_IMB_BUY : Market Imbalance - Buy. A 50,000 share or more excess of market orders to buy over market orders to sell as of 9:00am on expiration days. A MktImbBuy implies an ordinary order imbalance (OrdImbBuy).

• SECURITY_STATUS_QUAL_MKT_IMB_SELL : Market Imbalance - Sell. A 50,000 share or more excess of market orders to sell over market orders to buy as of 9:00am on expiration days. A MktImbSell implies an ordinary order imbalance (OrdImbSell).

• SECURITY_STATUS_QUAL_NO_MKT_IMB : No Market Imbalance. Indicates that the imbalance of market orders for a security is less than 50,000 shares as of 9:00am on expiration days.

• SECURITY_STATUS_QUAL_MOC_IMB_BUY : Market On Close Imbalance - Buy. An excess of 50,000 share or more of MOC orders to buy over MOC orders to sell (including MOC sell plus and MOC sell short orders). A MocImbBuy implies an ordinary order imbalance (OrdImbBuy).

• SECURITY_STATUS_QUAL_MOC_IMB_SELL : Market On Close Imbalance - Sell. An excess of 50,000 share or more of MOC orders to sell over orders to buy (including MOC orders to buy minus). A MocImbSell implies an ordinary order imbalance (OrdImbSell).

• SECURITY_STATUS_QUAL_NO_MOC_IMB : No Market On Close (MOC) Imbalance. The difference between the number of shares to buy MOC and the number of shares to sell MOC is less than 50,000.

• SECURITY_STATUS_QUAL_ORDER_IMB : Non-regulatory condition: a significant imbalance of buy orders exists for this security.

• SECURITY_STATUS_QUAL_ORDER_IMB_SELL : Non-regulatory condition: a significant imbalance of sell orders exists for this security.

• SECURITY_STATUS_QUAL_ORDER_OMB_NONE : The earlier imbalance of buy or sell orders no longer exists for this security. It also might mean that there is no imbalance to begin with.

• SECURITY_STATUS_QUAL_RANGE_IND : Trading Range Indication. Not an Opening Delay or Trading Halted condition: this condition is used prior to the opening of a security to denote a probable trading range (bid and offer prices, no sizes).

• SECURITY_STATUS_QUAL_ITS_PRE_OPEN : ITS pre-opening indication.

• SECURITY_STATUS_QUAL_RESERVED : Reserved (e.g., CME).
- SECURITY_STATUS_QUAL_FROZEN : Frozen (e.g., CME).
- SECURITY_STATUS_QUAL_PRE_OPEN : Preopening state (e.g., CME).
- SECURITY_STATUS_QUAL_ADD_INFO : Additional Information. For a security that is Opening Delayed or Trading Halted, if inadequate information is disclosed during a "news dissemination or news pending" Opening Delay or Trading Halt, the Opening Delay or Trading Halt reason could be subsequently reported as "Additional Information."
- SECURITY_STATUS_QUAL_OPEN_DELAY : Security’s opening has been delayed by exchange. This value is usually followed by another value specifying the reason for the opening delay.
- SECURITY_STATUS_QUAL_NO_OPEN_RESUME : Indicates that trading halt or opening delay will be in effect for the remainder of the trading day.
- SECURITY_STATUS_QUAL_PRICE_IND : An approximation of what a security’s opening or re-opening price range (bid and offer prices, no sizes) will be when trading resumes after a delayed opening or after a trading halt.
- SECURITY_STATUS_QUAL_EQUIPMENT : Non-regulatory condition: the ability to trade this security by a participant is temporarily inhibited due to a systems, equipment or communications facility problem, or for other technical reasons.
- SECURITY_STATUS_QUAL_FILINGS : Not current in regulatory filings.
- SECURITY_STATUS_QUAL_NEWS : News pending. Denotes a regulatory trading halt due to an expected news announcement which influence the security. An Opening Delay or Trading Halt may be continued once the news has been disseminated.
- SECURITY_STATUS_QUAL_NEWS_DISSEM : News Dissemination. Denotes a regulatory trading halt when relevant news influencing the security is being disseminated. Trading is suspended until the primary market determines that an adequate publication or disclosure of information has occurred.
- SECURITY_STATUS_QUAL_LISTING : Listing Noncompliance.
- SECURITY_STATUS_QUAL_OPERATION : Operational Halt
- SECURITY_STATUS_QUAL_INFO : Information Requested. Regulatory condition: more disclosure of information is requested by the exchange for this security.
- SECURITY_STATUS_QUAL_SEC : SEC Suspension.
- SECURITY_STATUS_QUAL_TIMES : News Resumption Times.
- SECURITY_STATUS_QUAL_OTHER : Other Regulatory Halt.
• SECURITY_STATUS_QUAL_RELATED : Related Security; In View Of Common. Non-regulatory condition: the halt or opening delay in this security is due to its relationship with another security. This condition also applies to non-common associated securities (e.g. warrants, rights, preferreds, classes, etc.) in view of the common stock.

• SECURITY_STATUS_QUAL_IPO : Upcoming IPO issue not yet trading.

Returns:

The normalized security status qualifier.

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.18 virtual MamdaFieldState Wombat::MamdaSecStatusRecap::getSecurityStatusQualifierEnumFieldState () const  [pure virtual]

Returns:

The normalized security status qualifier field state

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.19 virtual const char ∗ Wombat::MamdaSecStatusRecap::getSecurityStatusQualStr () const  [pure virtual]

Security status qualifier.

• None : No security status qualifier is known/available for this security.

• Excused : An Excused withdrawl from the market.

• Withdrawn : Non-excused withdrawl by the market maker.

• Suspended : Suspended Trading

• Resume : Resume trading/quoting after halt.

• QuoteResume : Resume quoting after halt. Nasdaq distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.

• TradeResume : Resume trading after halt. Nasdaq distinguishes between resumption of quoting versus trading, although these appear to always occur one after the other.
• ResumeTime : When the security is expected to resume trading.

• MktImbBuy : Market Imbalance - Buy. A 50,000 share or more excess of market orders to buy over market orders to sell as of 9:00am on expiration days. A MktImbBuy implies an ordinary order imbalance (OrdImbBuy).

• MktImbSell : Market Imbalance - Sell. A 50,000 share or more excess of market orders to sell over market orders to buy as of 9:00am on expiration days. A MktImbSell implies an ordinary order imbalance (OrdImbSell).

• NoMktImb : No Market Imbalance. Indicates that the imbalance of market orders for a security is less than 50,000 shares as of 9:00am on expiration days.

• MocImbBuy : Market On Close Imbalance - Buy. An excess of 50,000 share or more of MOC orders to buy over MOC orders to sell (including MOC sell plus and MOC sell short orders). A MocImbBuy implies an ordinary order imbalance (OrdImbBuy).

• MocImbSell : Market On Close Imbalance - Sell. An excess of 50,000 share or more of MOC orders to sell (not including MOC sell short and MOC sell plus orders) over orders to buy (including MOC orders to buy minus). A MocImbSell implies an ordinary order imbalance (OrdImbSell).

• NoMocImb : No Market On Close (MOC) Imbalance. The difference between the number of shares to buy MOC and the number of shares to sell MOC is less than 50,000.

• OrderImb : Non-regulatory condition: a significant imbalance of buy or sell orders exists for this security.

• OrderInf : Non-regulatory condition where there is a significant influx of orders.

• OrderImbBuy : Non-regulatory condition: a significant imbalance of buy orders exists for this security.

• OrderImbSell : Non-regulatory condition: a significant imbalance of sell orders exists for this security.

• OrderImbNone : The earlier imbalance of buy or sell orders no longer exists for this security. It also might mean that there is no imbalance to begin with.

• RangeInd : Trading Range Indication. Not an Opening Delay or Trading Halted condition: this condition is used prior to the opening of a security to denote a probable trading range (bid and offer prices, no sizes).

• ItsPreOpen : ITS pre-opening indication.

• Reserved : Reserved (e.g., CME).

• Frozen : Frozen (e.g., CME).
• PreOpen : Preopening state (e.g., CME).
• AddInfo : Additional Information. For a security that is Opening Delayed or Trading Halted, if inadequate information is disclosed during a “news dissemination or news pending” Opening Delay or Trading Halt, the Opening Delay or Trading Halt reason could be subsequently reported as "Additional Information."
• OpenDelay : Security’s opening has been delayed by exchange. This value is usually followed by another value specifying the reason for the opening delay.
• NoOpenNoResume : Indicates that trading halt or opening delay will be in effect for the remainder of the trading day.
• PriceInd : An approximation of what a security’s opening or re-opening price range (bid and offer prices, no sizes) will be when trading resumes after a delayed opening or after a trading halt.
• Equipment : Non-regulatory condition: the ability to trade this security by a participant is temporarily inhibited due to a systems, equipment or communications facility problem, or for other technical reasons.
• Filings : Not current in regulatory filings.
• News : News pending. Denotes a regulatory trading halt due to an expected news announcement which influence the security. An Opening Delay or Trading Halt may be continued once the news has been disseminated.
• NewsDissem : News Dissemination. Denotes a regulatory trading halt when relevant news influencing the security is being disseminated. Trading is suspended until the primary market determines that an adequate publication or disclosure of information has occurred.
• Listing : Listing Noncompliance.
• Operation : Operational Halt
• Info : Information Requested. Regulatory condition: more disclosure of information is requested by the exchange for this security.
• SEC : SEC Suspension.
• Times : News Resumption Times.
• Other : Other Regulatory Halt.
• Related : Related Security; In View Of Common. Non-regulatory condition: the halt or opening delay in this security is due to its relationship with another security. This condition also applies to non-common associated securities (e.g. warrants, rights, prefereds, classes, etc.) in view of the common stock.
• IPO : Upcoming IPO issue not yet trading.
Returns:

The normalized security status qualifier.

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.20 virtual MamdaFieldState Wombat::MamdaSecStatusRecap::getSecurityStatusQualStrFieldState () const [pure virtual]

Returns:

The normalized security status qualifier field state.

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.21 virtual const char * Wombat::MamdaSecStatusRecap::getSecurityStatusNative () const [pure virtual]

Original "security status" field sent by the feed.
NYSE Technologies also sends normalized security status.

Returns:

The exchange provided security status.

See also:

getSecurityStatus()

Implemented in Wombat::MamdaSecStatusListener.

6.130.3.22 virtual MamdaFieldState Wombat::MamdaSecStatusRecap::getSecurityStatusNativeFieldState () const [pure virtual]

Returns:

The exchange provided security status field state.

Implemented in Wombat::MamdaSecStatusListener.
virtual const char* Wombat::MamdaSecStatusRecap::getFreeText() const  [pure virtual]

Arbitrary free text associated with the security status change.

Returns:

Free text associated with the security status change.

Implemented in Wombat::MamdaSecStatusListener.

virtual MamdaFieldState Wombat::MamdaSecStatusRecap::getFreeTextFieldState() const  [pure virtual]

Returns:

Free text associated with the security status change field state.

Implemented in Wombat::MamdaSecStatusListener.

The documentation for this class was generated from the following file:

- MamdaSecStatusRecap.h
Wombat::MamdaSecStatusSymbolSourceAdapter Class Reference

MamdaSecStatusSymbolSourceAdapter is a simple adapter class that can be added as a handler to the MamdaSecStatusListener turning it into a MamdaSymbolSource.

#include <MamdaSecStatusSymbolSourceAdapter.h>

Inheritance diagram for Wombat::MamdaSecStatusSymbolSourceAdapter:

**Public Member Functions**

- MamdaSecStatusSymbolSourceAdapter()
- virtual ~MamdaSecStatusSymbolSourceAdapter()  
- void addHandler(MamdaSymbolSourceHandler *handler)
  
  Register a handler implementing the MamdaSymbolSourceHandler interface.

- virtual const char * getSourcedSymbol() const
  
  Return the sourced symbol.

- virtual void onSecStatusRecap(MamdaSubscription *subscription, MamdaSecStatusListener &listener, const MamaMsg &msg, MamdaSecStatusRecap &recap)
  
  Implementation of MamdaSecStatusHandler interface.

- virtual void onSecStatusUpdate(MamdaSubscription *subscription, MamdaSecStatusListener &listener, const MamaMsg &msg)
  
  Method invoked when a security status update is received.

**6.131.1 Detailed Description**

MamdaSecStatusSymbolSourceAdapter is a simple adapter class that can be added as a handler to the MamdaSecStatusListener turning it into a MamdaSymbolSource.

Objects implementing the MamdaSymbolSource interface can register with this adapter and receive notification of newly announced symbols indirectly from the MamdaSecStatusListener via this adapter.
6.131.2 Constructor & Destructor Documentation

6.131.2.1 Wombat::MamdaSecStatusSymbolSourceAdapter::MamdaSecStatusSymbolSourceAdapter()

6.131.2.2 virtual Wombat::MamdaSecStatusSymbolSourceAdapter::~MamdaSecStatusSymbolSourceAdapter()
   [virtual]

6.131.3 Member Function Documentation

6.131.3.1 void Wombat::MamdaSecStatusSymbolSourceAdapter::addHandler(MamdaSymbolSourceHandler *handler)

Register a handler implementing the MamdaSymbolSourceHandler interface.

Parameters:

   handler The symbol source handler

6.131.3.2 virtual const char* Wombat::MamdaSecStatusSymbolSourceAdapter::getSourcedSymbol() const
   [virtual]

Return the sourced symbol.

Implementation of the MamdaSymbolSourceEvent interface

Returns:

   sourced symbol

Implements Wombat::MamdaSymbolSourceEvent.

6.131.3.3 virtual void Wombat::MamdaSecStatusSymbolSourceAdapter::onSecStatusRecap(MamdaSubscription *subscription,
   MamdaSecStatusListener &listener, const MamaMsg &msg,
   MamdaSecStatusRecap &recap) [virtual]

Implementation of MamdaSecStatusHandler interface.

Parameters:

   subscription The security status subscription.
Listener The security status listener.

msg The security status update message.

Implements Wombat::MamdaSecStatusHandler.

6.131.3.4 virtual void Wombat::MamdaSecStatusSymbolSourceAdapter::onSecStatusUpdate (MamdaSubscription * subscription,
MamdaSecStatusListener & listener, const MamaMsg & msg)
[virtual]

Method invoked when a security status update is received.

Parameters:

subscription The subscription which received the update
listener The listener which invoked this callback.
msg The MamaMsg that triggered this invocation.

Implements Wombat::MamdaSecStatusHandler.

The documentation for this class was generated from the following file:

- MamdaSecStatusSymbolSourceAdapter.h
A `MamdaSubscription` is used to register interest in a particular symbol and source.

```
#include <MamdaSubscription.h>
```

**Public Member Functions**

- `MamdaSubscription()`  
  Default constructor.

- `virtual ~MamdaSubscription()`  
  Destructor.

- `virtual MamdaSubscription * clone () const`  
  Clone this `MamdaSubscription`.

- `void create (MamaQueue *queue, MamaSource *source, const char *symbol, void *closure=NULL)`  
  Create and activate a subscription.

- `void destroy ()`  
  Destroy a subscription.

- `bool isActive () const`  
  Return whether subscription is active.

- `void setSource (MamaSource *source)`  
  Set the data source name.

- `void setSymbol (const char *symbol)`  
  Set the symbol.

- `void setQueue (MamaQueue *queue)`  
  Set the MAMA queue.

- `void setMdDataType (mamaMdDataType mdDataType)`  
  Set the market data type.

- `void setType (mamaSubscriptionType type)`  
  Set the subscription type.
void setServiceLevel (mamaServiceLevel serviceLevel, long serviceLevelOpt=0)
  
  Set the MAMA service level.

void setRequireInitial (bool require)
  
  Set whether an initial value is required.

void setTimeout (double timeout)
  
  Set the subscription timeout (in seconds).

void setRetries (int retries)
  
  Set the subscription retries.

void setClosure (void *closure)
  
  Set the closure.

void setGroupSizeHint (int groupSizeHint)
  
  Set the group size hint.

void setMamaSubscription (MamaSubscription *subscription)
  
  Set the MamaSubscription.

void addMsgListener (MamdaMsgListener *listener)
  
  Add a listener for regular messages.

void addQualityListener (MamdaQualityListener *listener)
  
  Add a listener for changes in quality status.

void addErrorListener (MamdaErrorListener *listener)
  
  Add a listener for error events.

std::vector<MamdaMsgListener *> & getMsgListeners ()
  
  Return the vector of message listeners.

void activate ()
  
  Activate the subscription.

void deactivate ()
  
  Deactivate the subscription.

void requestRecap ()
  
  Deprecated.
• MamaSource * getSource () const
  
  Return the source.

• const char * getSourceName () const
  
  Return the publisher source name.

• const char * getSymbol () const
  
  Return the symbol.

• const char * getExchange () const
  
  Return the exchange.

• MamaTransport * getTransport () const
  
  Return the transport.

• MamaQueue * getQueue () const
  
  Return the queue.

• MamaSubscription * getMamaSubscription ()
  
  Return the MamaSubscription object.

• mamaSubscriptionType getType () const
  
  Return the subscription type.

• mamaServiceLevel getServiceLevel () const
  
  Return the service level.

• long getServiceLevelOpt () const
  
  Return the service level option.

• bool getRequireInitial () const
  
  Return whether an initial is required.

• double getTimeout () const
  
  Return the timeout (seconds).

• int getRetries () const
  
  Return the retries.

• void * getClosure () const
  
  Get the additional object passed as the closure to the create() method.
6.132 Wombat::MamdaSubscription Class Reference

- uint32_t getSeqNum () const
  
  Get the message-level sequence number.

- void setItemClosure (void *closure)
  
  Set the item closure for group subscriptions.

- void * getItemClosure (void)
  
  Get the item closure for group subscriptions.

- bool checkDebugLevel (MamaLogLevel level) const
  
  Return whether the debug level for this subscription equals or exceeds some level.

6.132.1 Detailed Description

A MamdaSubscription is used to register interest in a particular symbol and source. A MamaSource object is required to actually activate the subscription. Multiple listeners can be added to the MamdaSubscription. In this way, an application can make use of more than one of the specialized value added MAMDA listeners, such as MamdaTradeListener and MamdaQuoteListener.

The queue argument may be null to use Mama’s internal queue.

6.132.2 Constructor & Destructor Documentation

6.132.2.1 Wombat::MamdaSubscription::MamdaSubscription ()

Default constructor.
Use the create() method to create and activate the subscription.

6.132.2.2 virtual Wombat::MamdaSubscription::~MamdaSubscription ()

Destructor.
6.132.3 Member Function Documentation

6.132.3.1 virtual MamdaSubscription* Wombat::MamdaSubscription::clone () const [virtual]

Clone this MamdaSubscription.
Allocate an inactive MamdaSubscription object that is initialized as a duplicate of this one. The purpose of this facility is to enable a "template" of a MamdaSubscription for use with many subscriptions. The following subscription attributes are copied: source, symbol, queue, subscription type, service level, requires initial and timeout.

6.132.3.2 void Wombat::MamdaSubscription::create (MamaQueue * queue, MamaSource * source, const char * symbol, void * closure = NULL)

Create and activate a subscription.
Set any subscription properties prior to calling this method.

6.132.3.3 void Wombat::MamdaSubscription::destroy ()

Destroy a subscription.

6.132.3.4 bool Wombat::MamdaSubscription::isActive () const

Return whether subscription is active.

6.132.3.5 void Wombat::MamdaSubscription::setSource (MamaSource * source)

Set the data source name.
Do this before calling activate().

6.132.3.6 void Wombat::MamdaSubscription::setSymbol (const char * symbol)

Set the symbol.
Do this before calling activate().

6.132.3.7 void Wombat::MamdaSubscription::setQueue (MamaQueue * queue)

Set the MAMA queue.
Do this before calling `activate()`.

### 6.132.3.8 void Wombat::MamdaSubscription::setMdDataType

```cpp
(mamaMdDataType mdDataType)
```

Set the market data type.
Do this before calling `activate()`.

### 6.132.3.9 void Wombat::MamdaSubscription::setType

```cpp
(mamaSubscriptionType type)
```

Set the subscription type.
Do this before calling `activate()`.

### 6.132.3.10 void Wombat::MamdaSubscription::setServiceLevel

```cpp
(mamaServiceLevel serviceLevel, long serviceLevelOpt = 0)
```

Set the MAMA service level.

### 6.132.3.11 void Wombat::MamdaSubscription::setRequireInitial

```cpp
(bool require)
```

Set whether an initial value is required.
Do this before calling `activate()`.

### 6.132.3.12 void Wombat::MamdaSubscription::setTimeout

```cpp
(double timeout)
```

Set the subscription timeout (in seconds).
Do this before calling `activate()`.

### 6.132.3.13 void Wombat::MamdaSubscription::setRetries

```cpp
(int retries)
```

Set the subscription retries.
Do this before calling `activate()`.

### 6.132.3.14 void Wombat::MamdaSubscription::setClosure

```cpp
(void * closure)
```

Set the closure.
Do this before calling `activate()`.
6.132.3.15 void Wombat::MamdaSubscription::setGroupSizeHint (int groupSizeHint)

Set the group size hint.
Do this before calling activate().

6.132.3.16 void Wombat::MamdaSubscription::setMamaSubscription (MamaSubscription * subscription)

Set the MamaSubscription.
This is normally done automatically.

6.132.3.17 void Wombat::MamdaSubscription::addMsgListener (MamdaMsgListener * listener)

Add a listener for regular messages.

6.132.3.18 void Wombat::MamdaSubscription::addQualityListener (MamdaQualityListener * listener)

Add a listener for changes in quality status.

6.132.3.19 void Wombat::MamdaSubscription::addErrorListener (MamdaErrorListener * listener)

Add a listener for error events.

6.132.3.20 std::vector<MamdaMsgListener*>& Wombat::MamdaSubscription::getMsgListeners ()

Return the vector of message listeners.

Returns:
Vector of message listeners registered with the object.

6.132.3.21 void Wombat::MamdaSubscription::activate ()

Activate the subscription.
Until this method is invoked, no updates will be received. The parameters for the subscription should have been specified using the "set" methods.

6.132.3.22 void Wombat::MamdaSubscription::deactivate ()

Deactivate the subscription.
No more updates will be received for this subscription (unless activate() is invoked again).
This function must be called from the same thread dispatching on the associated event queue unless both the default queue and dispatch queue are not actively dispatching.

6.132.3.23 void Wombat::MamdaSubscription::requestRecap ()

 Deprecated.
This method is now a no-op.

6.132.3.24 MamaSource* Wombat::MamdaSubscription::getSource () const
Return the source.
Note: When using managed subscriptions this currently returns NULL

6.132.3.25 const char* Wombat::MamdaSubscription::getSourceName () const
Return the publisher source name.

6.132.3.26 const char* Wombat::MamdaSubscription::getSymbol () const
Return the symbol.

6.132.3.27 const char* Wombat::MamdaSubscription::getExchange () const
Return the exchange.

6.132.3.28 MamaTransport* Wombat::MamdaSubscription::getTransport () const
Return the transport.
6.132.3.29 MamaQueue* Wombat::MamdaSubscription::getQueue() const

Return the queue.

6.132.3.30 MamaSubscription* Wombat::MamdaSubscription::getMamaSubscription()

Return the MamaSubscription object.

6.132.3.31 mamaSubscriptionType Wombat::MamdaSubscription::getType() const

Return the subscription type.

6.132.3.32 mamaServiceLevel Wombat::MamdaSubscription::getServiceLevel() const

Return the service level.

6.132.3.33 long Wombat::MamdaSubscription::getServiceLevelOpt() const

Return the service level option.

6.132.3.34 bool Wombat::MamdaSubscription::getRequireInitial() const

Return whether an initial is required.

6.132.3.35 double Wombat::MamdaSubscription::getTimeout() const

Return the timeout (seconds).

6.132.3.36 int Wombat::MamdaSubscription::getRetries() const

Return the retries.

6.132.3.37 void* Wombat::MamdaSubscription::getClosure() const

Get the additional object passed as the closure to the create() method.
6.132.3.38 uint32_t Wombat::MamdaSubscription::getSeqNum () const

Get the message-level sequence number.
This number is normally sequential although there are some exceptions. Erroneous exceptions are reported via the "quality listener" interface.

6.132.3.39 void Wombat::MamdaSubscription::setItemClosure (void * closure)

Set the item closure for group subscriptions.
Setting the item closure for a non-group subscription provides a second closure.

6.132.3.40 void* Wombat::MamdaSubscription::getItemClosure (void)

Get the item closure for group subscriptions.

6.132.3.41 bool Wombat::MamdaSubscription::checkDebugLevel (MamaLogLevel level) const

Return whether the debug level for this subscription equals or exceeds some level.

Parameters:

level The debug level to check.

Returns:

whether the level equals or exceeds the set level for this subscription.

The documentation for this class was generated from the following file:

• MamdaSubscription.h
MamdaSymbolSourceEvent is an interface that provides access to a sourced symbol name.

#include <MamdaSymbolSourceEvent.h>

Inheritance diagram for Wombat::MamdaSymbolSourceEvent::

```
Wombat::MamdaSymbolSourceEvent
```

Wombat::MamdaSecStatusSymbolSourceAdapter

Public Member Functions

- virtual const char∗ getSourcedSymbol () const =0

  Get the sourced symbol.

- virtual ~MamdaSymbolSourceEvent ()

6.133.1 Detailed Description

MamdaSymbolSourceEvent is an interface that provides access to a sourced symbol name.

6.133.2 Constructor & Destructor Documentation

6.133.2.1 virtual Wombat::MamdaSymbolSourceEvent::~MamdaSymbolSourceEvent () [virtual]

45 {};

6.133.3 Member Function Documentation

6.133.3.1 virtual const char∗ Wombat::MamdaSymbolSourceEvent::getSourcedSymbol () const [pure

virtual]

Get the sourced symbol.
Returns:

The sourced symbol.

Implemented in Wombat::MamdaSecStatusSymbolSourceAdapter.

The documentation for this class was generated from the following file:

- MamdaSymbolSourceEvent.h
MamdaSymbolSourceHandler is an interface for applications that want to have an easy way to handle newly sourced symbol events.

```cpp
#include <MamdaSymbolSourceHandler.h>
```

### Public Member Functions

- `virtual void onSymbol (const MamdaSymbolSourceEvent &event)=0`  
  *Method invoked when a new symbol has been sourced.*

- `virtual ~MamdaSymbolSourceHandler ()`

### Detailed Description

MamdaSymbolSourceHandler is an interface for applications that want to have an easy way to handle newly sourced symbol events.

The interface defines the callback method for the new symbol event: onSymbol

### Constructor & Destructor Documentation

6.134.2.1 virtual Wombat::MamdaSymbolSourceHandler::~MamdaSymbolSourceHandler () [virtual]

* {};

### Member Function Documentation

6.134.3.1 virtual void Wombat::MamdaSymbolSourceHandler::onSymbol (const MamdaSymbolSourceEvent & event) [pure virtual]

Method invoked when a new symbol has been sourced.

**Parameters:**

- `event` The MamdaSymbolSourceEvent object.

The documentation for this class was generated from the following file:

- MamdaSymbolSourceHandler.h
Wombat::MamdaTradeCancelOrError Class Reference

MamdaTradeCancelOrError is an interface that provides access to trade cancellation related fields.

#include <MamdaTradeCancelOrError.h>

Inheritance diagram for Wombat::MamdaTradeCancelOrError:

Wombat::MamdaBasicEvent

Wombat::MamdaTradeCancelOrError

Wombat::MamdaTradeListener

Public Member Functions

• virtual bool getIsCancel () const =0
  Return whether this event is a trade cancel.

• virtual mama_seqnum_t getOrigSeqNum () const =0
  Original feed-generated sequence for a correction/cancel/error.

• virtual MamdaFieldState getOrigSeqNumFieldState () const =0
  Get the field state.

• virtual const MamaPrice & getOrigPrice () const =0
  Original trade price in a correction/cancel/error.

• virtual MamdaFieldState getOrigPriceFieldState () const =0
  Get the field state.

• virtual mama_quantity_t getOrigVolume () const =0
  Original trade size in a correction/cancel/error.

• virtual MamdaFieldState getOrigVolumeFieldState () const =0
  Get the field state.

• virtual const char * getOrigPartId () const =0
Original trade participant identifier in a correction/cancel/error.

- virtual MamdaFieldState getOrigPartIdFieldState () const =0
  Get the field state.

- virtual const char ∗ getOrigQual () const =0
  A normalized set of qualifiers for the original trade for the security in a correction/cancel/error.

- virtual MamdaFieldState getOrigQualFieldState () const =0
  Get the field state.

- virtual const char ∗ getOrigQualNative () const =0
  Feed-specific trade qualifier code(s) for original trade.

- virtual MamdaFieldState getOrigQualNativeFieldState () const =0
  Get the field state.

- virtual mama_u32_t getOrigSellersSaleDays () const =0
  Seller’s sale days for original trade.

- virtual MamdaFieldState getOrigSellersSaleDaysFieldState () const =0
  Get the field state.

- virtual char getOrigStopStock () const =0
  Stopped stock indicator for original trade.

- virtual MamdaFieldState getOrigStopStockFieldState () const =0
  Get the field state.

- virtual const char ∗ getTradeQual () const =0
  Get the wombat normalized trade qualifier.

- virtual MamdaFieldState getTradeQualFieldState () const =0
  The trade qual Field State.

- virtual bool getIsIrregular () const =0
  Get whether the trade is irregular.

- virtual MamdaFieldState getIsIrregularFieldState () const =0
  The isIrregular Field State.
6.135 Wombat::MamdaTradeCancelOrError Class Reference

• virtual MamdaFieldState getGenericFlagFieldState () const =0
• virtual const char * getOrigTradeId () const =0
  Get the original trade id.

• virtual MamdaFieldState getOrigTradeIdFieldState () const =0
• virtual char getOrigShortSaleCircuitBreaker () const =0
  get the OrigShortSaleCircuitBreaker

• virtual MamdaFieldState getOrigShortSaleCircuitBreakerFieldState () const =0
• virtual ~MamdaTradeCancelOrError ()

6.135.1 Detailed Description

MamdaTradeCancelOrError is an interface that provides access to trade cancellation related fields.

6.135.2 Constructor & Destructor Documentation

6.135.2.1 virtual Wombat::MamdaTradeCancelOrError::~MamdaTradeCancelOrError () [virtual]

448 {};

6.135.3 Member Function Documentation

6.135.3.1 virtual bool Wombat::MamdaTradeCancelOrError::getIsCancel () const [pure virtual]

Return whether this event is a trade cancel.
If false, the event is a trade error.

Returns:
  Whether this is a trade cancel.

Implemented in Wombat::MamdaTradeListener.

6.135.3.2 virtual mama_seqnum_t Wombat::MamdaTradeCancelOrError::getOrigSeqNum () const [pure virtual]

Original feed-generated sequence for a correction/cancel/error.
Returns:
The original sequence number.

Implemented in Wombat::MamdaTradeListener.

6.135.3.3 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getOrigSeqNumFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.135.3.4 virtual const MamaPrice& Wombat::MamdaTradeCancelOrError::getOrigPrice () const [pure virtual]

Original trade price in a correction/cancel/error.

Returns:
The original trade price.

Implemented in Wombat::MamdaTradeListener.

6.135.3.5 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getOrigPriceFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.135.3.6 virtual mama_quantity_t Wombat::MamdaTradeCancelOrError::getOrigVolume () const [pure virtual]

Original trade size in a correction/cancel/error.

**Returns:**

The original trade volume.

Implemented in `Wombat::MamdaTradeListener`.

6.135.3.7 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getOrigVolumeFieldState () const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaTradeListener`.

6.135.3.8 virtual const char* Wombat::MamdaTradeCancelOrError::getOrigPartId () const [pure virtual]

Original trade participant identifier in a correction/cancel/error.

**Returns:**

The original trade participant identifier.

Implemented in `Wombat::MamdaTradeListener`.

6.135.3.9 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getOrigPartIdFieldState () const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaTradeListener`.
6.135.3.10 virtual const char* Wombat::MamdaTradeCancelOrError::getOrigQual () const [pure virtual]

A normalized set of qualifiers for the original trade for the security in a correction/cancel/error.

Returns:
The original trade qualifier.

Implemented in Wombat::MamdaTradeListener.

6.135.3.11 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getOrigQualFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.135.3.12 virtual const char* Wombat::MamdaTradeCancelOrError::getOrigQualNative () const [pure virtual]

Feed-specific trade qualifier code(s) for original trade.
This field is provided primarily for completeness and/or troubleshooting.

Returns:
The original trade condition.

Implemented in Wombat::MamdaTradeListener.

6.135.3.13 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getOrigQualNativeFieldState () const [pure virtual]

Get the field state.
Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.135.3.14 virtual mama_u32_t Wombat::MamdaTradeCancelOrError::getOrigSellersSaleDays () const [pure virtual]

Seller’s sale days for original trade.
Used when the trade qualifier is "Seller". Specifies the number of days that may elapse before delivery of the security.

Returns:
The original seller’s sale days.

Implemented in Wombat::MamdaTradeListener.

6.135.3.15 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getOrigSellersSaleDaysFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.135.3.16 virtual char Wombat::MamdaTradeCancelOrError::getOrigStopStock () const [pure virtual]

Stopped stock indicator for original trade.
Condition related to certain NYSE trading rules. This is not related to a halted security status. (0 == N/A; 1 == Applicable)

Returns:
The original stopped stock indicator.

Implemented in Wombat::MamdaTradeListener.
6.135.3.17 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getOrigStopStockFieldState () const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.135.3.18 virtual const char* Wombat::MamdaTradeCancelOrError::getTradeQual () const [pure virtual]

Get the wombat normalized trade qualifier.

Returns:

Trade qualifier. A normalized set of qualifiers for the current trade for the security. This field may contain multiple string values, separated by the colon(:) character.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Regular trade. A trade made without stated conditions is deemed regular way for settlement on the third * business day following the transaction * date.</td>
</tr>
<tr>
<td>Acquisition</td>
<td>A transaction made on the Exchange as a result of an Exchange acquisition.</td>
</tr>
<tr>
<td>Bunched</td>
<td>A trade representing an aggregate of two or more regular trades in a security occurring at the same price either simultaneously or within the same 60 second period, with no individual trade exceeding 10,000 shares.</td>
</tr>
<tr>
<td>Cash</td>
<td>A transaction which calls for the delivery of securities and payment on the same day the trade takes place.</td>
</tr>
<tr>
<td>Distribution</td>
<td>Sale of a large block of stock in such a manner that the price is not adversely affected.</td>
</tr>
<tr>
<td>BunchedSold</td>
<td>A bunched trade which is reported late</td>
</tr>
<tr>
<td>Rule155</td>
<td>To qualify as a 155 print, a specialist arranges for the sale of the block at one &quot;clean-up&quot; price or at the different price limits on his book. If the block is sold at a &quot;clean-up&quot; price, the specialist should execute at the same price all the executable buy orders on his book. The sale qualifier is only applicable for AMEX trades.</td>
</tr>
<tr>
<td>SoldLast</td>
<td>Sold Last is used when a trade prints in sequence but is reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>NextDay</td>
<td>A transaction which calls for delivery of securities on the first business day after the trade date.</td>
</tr>
<tr>
<td>Opened</td>
<td>Indicates an opening transaction that is printed out of sequence or reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>PriorRef</td>
<td>An executed trade that relates to an obligation to trade at an earlier point in the trading day or that refer to a prior reference price. This may be the result of an order that was lost or misplaced or a SelectNet order that was not executed on a timely basis.</td>
</tr>
<tr>
<td>SplitTrade</td>
<td>An execution in two markets when the specialist or Market Maker in the market first receiving the order across a wrapper.</td>
</tr>
</tbody>
</table>
6.135.3.19 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getTradeQualFieldState () const [pure virtual]

The trade qual Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.135.3.20 virtual bool Wombat::MamdaTradeCancelOrError::getIsIrregular () const [pure virtual]

Get whether the trade is irregular.

Returns:

Whether or not the trade qualifies as an irregular trade. In general, only "regular" trades qualify to update the official last price and high/low prices.

Implemented in Wombat::MamdaTradeListener.

6.135.3.21 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getIsIrregularFieldState () const [pure virtual]

The isIrregular Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.135.3.22 virtual bool Wombat::MamdaTradeCancelOrError::getGenericFlag () const [pure virtual]

Implemented in Wombat::MamdaTradeListener.
6.135.3.23 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getGenericFlagFieldState () const [pure virtual]

Implemented in Wombat::MamdaTradeListener.

6.135.3.24 virtual const char ∗ Wombat::MamdaTradeCancelOrError::getOrigTradeId () const [pure virtual]

Get the original trade id.

Returns:

the original trade id

Implemented in Wombat::MamdaTradeListener.

6.135.3.25 virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getOrigTradeIdFieldState () const [pure virtual]

Implemented in Wombat::MamdaTradeListener.

6.135.3.26 virtual char Wombat::MamdaTradeCancelOrError::getOrigShortSaleCircuitBreaker () const [pure virtual]

get the OrigShortSaleCircuitBreaker

Returns:

OrigShortSaleCircuitBreaker

• return values:
  • Blank: Short Sale Restriction Not in Effect.
  • A: Short Sale Restriction Activated.
  • C: Short Sale Restriction Continued.
  • D: Sale Restriction Deactivated.
  • E: Sale Restriction in Effect.

Implemented in Wombat::MamdaTradeListener.
virtual MamdaFieldState Wombat::MamdaTradeCancelOrError::getOrigShortSaleCircuitBreakerFieldState () const [pure virtual]

**Returns:**

The OrigShortSaleCircuitBreaker Field State.

Implemented in `Wombat::MamdaTradeListener`.

The documentation for this class was generated from the following file:

- `MamdaTradeCancelOrError.h`
MamdaTradeChecker is a class that provides trades sanity checking by periodically requesting snapshots of the trades from the publisher and comparing that with an trades being maintained in real time.

```
#include <MamdaTradeChecker.h>
```

### Public Member Functions

- **MamdaTradeChecker** (MamaQueue *queue, MamdaCheckerHandler *handler, MamaSource *source, const char *symbol, mama_f64_t intervalSeconds)
  
  Constructor.

- **~MamdaTradeChecker** ()
  
  Destructor.

- **void checkSnapShotNow ()**
  
  Perform an ad hoc snapshot check now.

- **mama_u32_t getSuccessCount () const**
  
  Return the number of successful counts.

- **mama_u32_t getInconclusiveCount () const**
  
  Return the number of inconclusive counts.

- **mama_u32_t getFailureCount () const**
  
  Return the number of failed checks.

### Detailed Description

MamdaTradeChecker is a class that provides trades sanity checking by periodically requesting snapshots of the trades from the publisher and comparing that with an trades being maintained in real time.

This class is purely for testing purposes, to test for possible configuration or programming errors in the trades publisher and in MAMDA trades management code.

The developer registers a handler that contains callbacks for successful, inconclusive and failure events. The developer also provides an interval representing the frequency of the snapshot checks. The first check will take place at some random point in time between zero and the interval.
6.136.2 Constructor & Destructor Documentation

6.136.2.1 Wombat::MamdaTradeChecker::MamdaTradeChecker
(MamaQueue * queue, MamdaCheckerHandler * handler,
MamaSource * source, const char * symbol, mama_f64_t
intervalSeconds)

Constructor.

6.136.2.2 Wombat::MamdaTradeChecker::~MamdaTradeChecker ()

Destructor.

6.136.3 Member Function Documentation

6.136.3.1 void Wombat::MamdaTradeChecker::checkSnapShotNow ()

Perform an ad hoc snapshot check now.
This may be useful if the checking is to be performed by some external trigger event.

6.136.3.2 mama_u32_t Wombat::MamdaTradeChecker::getSuccessCount ()
const

Returns:
The number of successful checks.

6.136.3.3 mama_u32_t Wombat::MamdaTradeChecker::getInconclusiveCount ()
const

Return the number of inconclusive counts.
An attempt to check the order book may be inconclusive if the order book sequence numbers do not match up.
Returns:
The number of inconclusive checks.

6.136.3.4 mama_u32_t Wombat::MamdaTradeChecker::getFailureCount ()
const

Return the number of failed checks.
This should be zero, of course.

**Returns:**

The number of failed checks.

The documentation for this class was generated from the following file:

- *MamdaTradeChecker.h*
6.137 Wombat::MamdaTradeClosing Class Reference

MamdaTradeClosing is an interface that provides access to trade closing related fields.

#include <MamdaTradeClosing.h>

Inheritance diagram for Wombat::MamdaTradeClosing::

```
#include "MamdaTradeClosing.h"

Public Member Functions

• virtual const MamaPrice & getClosePrice () const =0
  Return the Close price Today’s closing price.

• virtual MamdaFieldState getClosePriceFieldState () const =0
  Get the field state.

• virtual bool getIsIndicative () const =0
  Return whether this closing price is indicative or official.

• virtual ~MamdaTradeClosing ()

6.137.1 Detailed Description

MamdaTradeClosing is an interface that provides access to trade closing related fields.

6.137.2 Constructor & Destructor Documentation

6.137.2.1 virtual Wombat::MamdaTradeClosing::~MamdaTradeClosing ()
  [virtual]

63 {};

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
6.137.3 Member Function Documentation

6.137.3.1 virtual const MamaPrice& Wombat::MamdaTradeClosing::getClosePrice () const [pure virtual]

Return the Close price Today's closing price.
The close price is populated when official closing prices are sent by the feed after the session close.

Returns:
The trade closing price.

Implemented in Wombat::MamdaTradeListener.

6.137.3.2 virtual MamdaFieldState Wombat::MamdaTradeClosing::getClosePriceFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.137.3.3 virtual bool Wombat::MamdaTradeClosing::getIsIndicative () const [pure virtual]

Return whether this closing price is indicative or official.

Returns:
Whether the closing price is indicative or otherwise.

Implemented in Wombat::MamdaTradeListener.

The documentation for this class was generated from the following file:

• MamdaTradeClosing.h
6.138  Wombat::MamdaTradeCorrection Class Reference

MamdaTradeCorrection is an interface that provides access to trade correction related fields.

#include <MamdaTradeCorrection.h>

Inheritance diagram for Wombat::MamdaTradeCorrection:

```
Wombat::MamdaBasicEvent
    Wombat::MamdaTradeCorrection
    Wombat::MamdaTradeListener
```

Public Member Functions

- virtual mama_seqnum_t getOrigSeqNum () const =0
  Get the original trade sequence number.

- virtual MamdaFieldState getOrigSeqNumFieldState () const =0
  Get the original trade sequence number field state.

- virtual const MamaPrice & getOrigPrice () const =0
  Get the original trade price.

- virtual MamdaFieldState getOrigPriceFieldState () const =0
  Get the original trade price field state.

- virtual mama_quantity_t getOrigVolume () const =0
  Get the original trade volume.

- virtual MamdaFieldState getOrigVolumeFieldState () const =0
  Get the original trade volume field state.

- virtual const char * getOrigPartId () const =0
  Get the original trade participant identifier.

- virtual MamdaFieldState getOrigPartIdFieldState () const =0
  Get the original trade participant identifier.
Get the field state.

- virtual const char * getOrigQual () const =0  
  Get original trade qualifier.

- virtual MamdaFieldState getOrigQualFieldState () const =0  
  Get the field state.

- virtual const char * getOrigQualNative () const =0
  Get original trade condition.

- virtual MamdaFieldState getOrigQualNativeFieldState () const =0
  Get the field state.

- virtual mama_u32_t getOrigSellersSaleDays () const =0
  Get the original trade sellers days.

- virtual MamdaFieldState getOrigSellersSaleDaysFieldState () const =0
  Get the field state.

- virtual char getOrigStopStock () const =0
  Get the original stock stop indicator.

- virtual MamdaFieldState getOrigStopStockFieldState () const =0
  Get the field state.

- virtual const char * getCorrTradeId () const =0
  Get the corrected trade Id.

- virtual MamdaFieldState getCorrTradeIdFieldState () const =0
  Get the field state.

- virtual const MamaPrice & getCorrPrice () const =0
  Get the corrected trade price.

- virtual MamdaFieldState getCorrPriceFieldState () const =0
  Get the field state.

- virtual mama_quantity_t getCorrVolume () const =0
  Get the corrected trade volume.

- virtual MamdaFieldState getCorrVolumeFieldState () const =0
Get the field state.

- virtual const char* getCorrPartId () const =0
  Get the corrected trade participant identifier.

- virtual MamdaFieldState getCorrPartIdFieldState () const =0
  Get the field state.

- virtual const char* getCorrQual () const =0
  Get corrected trade qualifier.

- virtual MamdaFieldState getCorrQualFieldState () const =0
  Get the field state.

- virtual const char* getCorrQualNative () const =0
  Get corrected trade condition.

- virtual MamdaFieldState getCorrQualNativeFieldState () const =0
  Get the field state.

- virtual mama_u32_t getCorrSellersSaleDays () const =0
  Get the corrected trade sellers days.

- virtual MamdaFieldState getCorrSellersSaleDaysFieldState () const =0
  Get the field state.

- virtual char getCorrStopStock () const =0
  Get the original stock stop indicator.

- virtual MamdaFieldState getCorrStopStockFieldState () const =0
  Get the field state.

- virtual const char* getTradeQual () const =0
  Get the NYSE Technologies normalized trade qualifier.

- virtual MamdaFieldState getTradeQualFieldState () const =0
  The trade qual Field State.

- virtual bool getIsIrregular () const =0
  Get whether the trade is irregular.

- virtual MamdaFieldState getIsIrregularFieldState () const =0
The isIrregular Field State.

- virtual bool getGenericFlag () const =0
- virtual MamdaFieldState getGenericFlagFieldState () const =0
- virtual const char ∗ getOrigTradeId () const =0

Get the original trade id.

- virtual MamdaFieldState getOrigTradeIdFieldState () const =0
- virtual char getCorrShortSaleCircuitBreaker () const =0
  get the CorrShortSaleCircuitBreaker

- virtual MamdaFieldState getCorrShortSaleCircuitBreakerFieldState () const =0
- virtual ~MamdaTradeCorrection ()

### 6.138.1 Detailed Description

MamdaTradeCorrection is an interface that provides access to trade correction related fields.

### 6.138.2 Constructor & Destructor Documentation

#### 6.138.2.1 virtual Wombat::MamdaTradeCorrection::~MamdaTradeCorrection () [virtual]

527();

### 6.138.3 Member Function Documentation

#### 6.138.3.1 virtual mama_seqnum_t Wombat::MamdaTradeCorrection::getOrigSeqNum () const [pure virtual]

Get the original trade sequence number.

See also:

MamdaBasicEvent::getEventSeqNum()

Implemented in Wombat::MamdaTradeListener.
6.138.3.2 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getOrigSeqNumFieldState () const [pure virtual]

Get the original trade sequence number field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.3 virtual const MamaPrice& Wombat::MamdaTradeCorrection::getOrigPrice () const [pure virtual]

Get the original trade price.

See also:
MamdaTradeReport::getTradePrice()

Implemented in Wombat::MamdaTradeListener.

6.138.3.4 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getOrigPriceFieldState () const [pure virtual]

Get the original trade price field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.5 virtual mama_quantity_t Wombat::MamdaTradeCorrection::getOrigVolume () const [pure virtual]

Get the original trade volume.

See also:
MamdaTradeReport::getTradeVolume()

Implemented in Wombat::MamdaTradeListener.
virtual MamdaFieldState Wombat::MamdaTradeCorrection::getOrigVolumeFieldState () const [pure virtual]

Get the original trade volume field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual const char∗ Wombat::MamdaTradeCorrection::getOrigPartId () const [pure virtual]

Get the original trade participant identifier.

See also:
MamdaTradeReport::getTradePartId()

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeCorrection::getOrigPartIdFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual const char∗ Wombat::MamdaTradeCorrection::getOrigQual () const [pure virtual]

Get original trade qualifier.

See also:
MamdaTradeReport::getTradeQual()

Implemented in Wombat::MamdaTradeListener.
6.138.3.10  virtual MamdaFieldState Wombat::MamdaTradeCorrection::getOrigQualFieldState () const  [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.11  virtual const char ∗ Wombat::MamdaTradeCorrection::getOrigQualNative () const  [pure virtual]

Get original trade condition.

See also:

MamdaTradeReport::getTradeCondition()

Implemented in Wombat::MamdaTradeListener.

6.138.3.12  virtual MamdaFieldState Wombat::MamdaTradeCorrection::getOrigQualNativeFieldState () const  [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.13  virtual mama_u32_t Wombat::MamdaTradeCorrection::getOrigSellersSaleDays () const  [pure virtual]

Get the original trade sellers days.

See also:

MamdaTradeReport::getTradeSellersSaleDays()

Implemented in Wombat::MamdaTradeListener.
6.138.3.14 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getOrigSellersSaleDaysFieldState () const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.15 virtual char Wombat::MamdaTradeCorrection::getOrigStopStock () const [pure virtual]

Get the original stock stop indicator.

**See also:**

MamdaTradeReport::getTradeStopStock()

Implemented in Wombat::MamdaTradeListener.

6.138.3.16 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getOrigStopStockFieldState () const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.17 virtual const char∗ Wombat::MamdaTradeCorrection::getCorrTradeId () const [pure virtual]

Get the corrected trade Id.

Implemented in Wombat::MamdaTradeListener.
6.138.3.18 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getCorrTradeIdFieldState () const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.19 virtual const MamaPrice& Wombat::MamdaTradeCorrection::getCorrPrice () const [pure virtual]

Get the corrected trade price.

**See also:**

MamdaTradeReport::getTradePrice()

Implemented in Wombat::MamdaTradeListener.

6.138.3.20 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getCorrPriceFieldState () const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.21 virtual mama_quantity_t Wombat::MamdaTradeCorrection::getCorrVolume () const [pure virtual]

Get the corrected trade volume.

**See also:**

MamdaTradeReport::getTradeVolume()

Implemented in Wombat::MamdaTradeListener.
6.138.3.22 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getCorrVolumeFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.23 virtual const char ∗ Wombat::MamdaTradeCorrection::getCorrPartId () const [pure virtual]

Get the corrected trade participant identifier.

See also:
MamdaTradeReport::getTradePartId()

Implemented in Wombat::MamdaTradeListener.

6.138.3.24 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getCorrPartIdFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.25 virtual const char ∗ Wombat::MamdaTradeCorrection::getCorrQual () const [pure virtual]

Get corrected trade qualifier.

See also:
MamdaTradeReport::getTradeQual()

Implemented in Wombat::MamdaTradeListener.
Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaTradeListener`.

---

**6.138.3.27** virtual const char∗ `Wombat::MamdaTradeCorrection::getCorrQualNative () const` [pure virtual]

Get corrected trade condition.

**See also:**

MamdaTradeReport::getTradeCondition()

Implemented in `Wombat::MamdaTradeListener`.

---

**6.138.3.28** virtual `MamdaFieldState Wombat::MamdaTradeCorrection::getCorrQualNativeFieldState () const` [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaTradeListener`.

---

**6.138.3.29** virtual `mama_u32_t Wombat::MamdaTradeCorrection::getCorrSellersSaleDays () const` [pure virtual]

Get the corrected trade sellers days.

**See also:**

MamdaTradeReport::getTradeSellersSaleDays()

Implemented in `Wombat::MamdaTradeListener`. 

---
virtual MamdaFieldState Wombat::MamdaTradeCorrection::getCorrSellersSaleDaysFieldState () const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual char Wombat::MamdaTradeCorrection::getCorrStopStock () const [pure virtual]

Get the original stock stop indicator.

See also:

MamdaTradeReport::getTradeStopStock()

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeCorrection::getCorrStopStockFieldState () const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual const char * Wombat::MamdaTradeCorrection::getTradeQual () const [pure virtual]

Get the NYSE Technologies normalized trade qualifier.

Returns:

Trade qualifier. A normalized set of qualifiers for the current trade for the security. This field may contain multiple string values, separated by the colon(;) character.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Regular trade. A trade made without stated conditions is deemed regular way for settlement on the third business day following the trade date.</td>
</tr>
<tr>
<td>Acquisition</td>
<td>A transaction made on the Exchange as a result of an Exchange acquisition.</td>
</tr>
<tr>
<td>Bunched</td>
<td>A trade representing an aggregate of two or more regular trades in a security occurring at the same price either simultaneously or within the same 60 second period, with no individual trade exceeding 10,000 shares.</td>
</tr>
<tr>
<td>Cash</td>
<td>A transaction which calls for the delivery of securities and payment on the same day the trade takes place.</td>
</tr>
<tr>
<td>Distribution</td>
<td>Sale of a large block of stock in such a manner that the price is not adversely affected.</td>
</tr>
<tr>
<td>BunchedSold</td>
<td>A bunched trade which is reported late for settlement on the third business day.</td>
</tr>
<tr>
<td>Rule155</td>
<td>To qualify as a 155 print, a specialist arranges for the sale of the block at one &quot;clean-up&quot; price or at the different price limits on his book. If the block is sold at a &quot;clean-up&quot; price, the specialist should execute at the same price all the executable buy orders on his book. The sale qualifier is only applicable for AMEX trades.</td>
</tr>
<tr>
<td>SoldLast</td>
<td>Sold Last is used when a trade prints in sequence but is reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>NextDay</td>
<td>A transaction which calls for delivery of securities on the first business day after the trade date.</td>
</tr>
<tr>
<td>Opened</td>
<td>Indicates an opening transaction that is printed out of sequence or reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>PriorRef</td>
<td>An executed trade that relates to an obligation to trade at an earlier point in the trading day or that refer to a prior reference price. This may be the result of an order that was lost or misplaced or a SelectNet order that was not executed on a timely basis.</td>
</tr>
<tr>
<td>Seller</td>
<td>A Seller's option transaction is a special transaction which gives the seller the right to deliver the stock at any time within a specific period, ranging from not less than four calendar days to no more than 60 calendar days.</td>
</tr>
<tr>
<td>SplitTrade</td>
<td>An execution in two markets when the specialist or Market Maker in the market first receiving the order agrees.</td>
</tr>
</tbody>
</table>
6.138 Wombat::MamdaTradeCorrection Class Reference

Implemented in Wombat::MamdaTradeListener.

6.138.3.34 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getTradeQualFieldState () const [pure virtual]

The trade qual Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.35 virtual bool Wombat::MamdaTradeCorrection::getIsIrregular () const [pure virtual]

Get whether the trade is irregular.

Returns:
Whether or not the trade qualifies as an irregular trade. In general, only "regular" trades qualify to update the official last price and high/low prices.

Implemented in Wombat::MamdaTradeListener.

6.138.3.36 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getIsIrregularFieldState () const [pure virtual]

The isIrregular Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.138.3.37 virtual bool Wombat::MamdaTradeCorrection::getGenericFlag () const [pure virtual]

Implemented in Wombat::MamdaTradeListener.
6.138.3.38 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getGenericFlagFieldState () const [pure virtual]

Implemented in Wombat::MamdaTradeListener.

6.138.3.39 virtual const char ∗ Wombat::MamdaTradeCorrection::getOrigTradeId () const [pure virtual]

Get the original trade id.

Returns:

the original trade id

Implemented in Wombat::MamdaTradeListener.

6.138.3.40 virtual MamdaFieldState Wombat::MamdaTradeCorrection::getOrigTradeIdFieldState () const [pure virtual]

Implemented in Wombat::MamdaTradeListener.

6.138.3.41 virtual char Wombat::MamdaTradeCorrection::getCorrShortSaleCircuitBreaker () const [pure virtual]

get the CorrShortSaleCircuitBreaker

Returns:

CorrShortSaleCircuitBreaker

- return values:
  - Blank: Short Sale Restriction Not in Effect.
  - A: Short Sale Restriction Activated.
  - C: Short Sale Restriction Continued.
  - D: Sale Restriction Deactivated.
  - E: Sale Restriction in Effect.

 Implemented in Wombat::MamdaTradeListener.
virtual MamdaFieldState Wombat::MamdaTradeCorrection::get-CorrShortSaleCircuitBreakerFieldState () const  [pure virtual]

Returns:

The OrigShortSaleCircuitBreaker Field State.

Implemented in Wombat::MamdaTradeListener.

The documentation for this class was generated from the following file:

- MamdaTradeCorrection.h
Utility cache of MamaFieldDescriptors which are used internally by the API when accessing trade related fields from update messages.

#include <MamdaTradeFields.h>

Static Public Member Functions

- static void setDictionary (const MamaDictionary &dictionary)
- static void reset ()
  Reset the dictionary for trade update fields.
- static uint16_t getMaxFid ()
- static bool isSet ()

Static Public Attributes

- static const MamaFieldDescriptor * TRADE_PRICE
- static const MamaFieldDescriptor * TRADE_DATE
- static const MamaFieldDescriptor * TRADE_SIDE
- static const MamaFieldDescriptor * AGGRESSOR_SIDE
- static const MamaFieldDescriptor * TRADE_TIME
- static const MamaFieldDescriptor * LAST_PRICE
- static const MamaFieldDescriptor * LAST_VOLUME
- static const MamaFieldDescriptor * LAST_DATE_TIME
- static const MamaFieldDescriptor * LAST_PART_ID
- static const MamaFieldDescriptor * LAST_DIRECTION
- static const MamaFieldDescriptor * NET_CHANGE
- static const MamaFieldDescriptor * PCT_CHANGE
- static const MamaFieldDescriptor * TRADE_SIZE
- static const MamaFieldDescriptor * TOTAL_VOLUME
- static const MamaFieldDescriptor * OFF_EXCHANGE_TOTAL_VOLUME
- static const MamaFieldDescriptor * ON_EXCHANGE_TOTAL_VOLUME
- static const MamaFieldDescriptor * TRADE_UNITS
- static const MamaFieldDescriptor * HIGH_PRICE
- static const MamaFieldDescriptor * LOW_PRICE
- static const MamaFieldDescriptor * OPEN_PRICE
- static const MamaFieldDescriptor * CLOSE_PRICE
- static const MamaFieldDescriptor * CLOSE_DATE
- static const MamaFieldDescriptor * PREV_CLOSE_PRICE
- static const MamaFieldDescriptor * PREV_CLOSE_DATE
static const MamaFieldDescriptor * ADJ_PREV_CLOSE
static const MamaFieldDescriptor * PREV_VOLUME
static const MamaFieldDescriptor * TRADE_SEQNUM
static const MamaFieldDescriptor * TRADE_QUALIFIER
static const MamaFieldDescriptor * TRADE_PART_ID
static const MamaFieldDescriptor * TOTAL_VALUE
static const MamaFieldDescriptor * OFF_EXCHANGE_TOTAL_VALUE
static const MamaFieldDescriptor * ON_EXCHANGE_TOTAL_VALUE
static const MamaFieldDescriptor * VWAP
static const MamaFieldDescriptor * OFF_EXCHANGE_VWAP
static const MamaFieldDescriptor * ON_EXCHANGE_VWAP
static const MamaFieldDescriptor * STD_DEV
static const MamaFieldDescriptor * STD_DEV_SUM
static const MamaFieldDescriptor * STD_DEV_SUM_SQUARES
static const MamaFieldDescriptor * SALE_CONDITION
static const MamaFieldDescriptor * SELLERS_SALE_DAYS
static const MamaFieldDescriptor * IS_IRREGULAR
static const MamaFieldDescriptor * IRREG_PART_ID
static const MamaFieldDescriptor * IRREG_PRICE
static const MamaFieldDescriptor * IRREG_SIZE
static const MamaFieldDescriptor * IRREG_TIME
static const MamaFieldDescriptor * ORIG_PART_ID
static const MamaFieldDescriptor * ORIG_PRICE
static const MamaFieldDescriptor * ORIG_SIZE
static const MamaFieldDescriptor * ORIG_SEQNUM
static const MamaFieldDescriptor * ORIG_TRADE_QUALIFIER
static const MamaFieldDescriptor * ORIG_SALE_CONDITION
static const MamaFieldDescriptor * ORIG_SELLERS_SALE_DAYS
static const MamaFieldDescriptor * ORIG_STOP_STOCK_IND
static const MamaFieldDescriptor * STOP_STOCK_IND
static const MamaFieldDescriptor * CORR_PART_ID
static const MamaFieldDescriptor * CORR_PRICE
static const MamaFieldDescriptor * CORR_SIZE
static const MamaFieldDescriptor * CORR_TRADE_QUALIFIER
static const MamaFieldDescriptor * CORR_SALE_CONDITION
static const MamaFieldDescriptor * CORR_TRADE_ID
static const MamaFieldDescriptor * CORR_SELLERS_SALE_DAYS
static const MamaFieldDescriptor * CORR_STOP_STOCK_IND
static const MamaFieldDescriptor * CORR_TIME
static const MamaFieldDescriptor * CANCEL_TIME
static const MamaFieldDescriptor * TRADE_ID
static const MamaFieldDescriptor * ORIG_TRADE_ID
static const MamaFieldDescriptor * PRIMARY_EXCH
• static const MamaFieldDescriptor * TRADE_COUNT
• static const MamaFieldDescriptor * BLOCK_COUNT
• static const MamaFieldDescriptor * BLOCK_VOLUME
• static const MamaFieldDescriptor * ORDER_ID
• static const MamaFieldDescriptor * UPDATE_AS_TRADE
• static const MamaFieldDescriptor * CURRENCY_CODE
• static const MamaFieldDescriptor * SETTLE_PRICE
• static const MamaFieldDescriptor * SETTLE_DATE
• static const MamaFieldDescriptor * HIGH_SEQNUM
• static const MamaFieldDescriptor * LOW_SEQNUM
• static const MamaFieldDescriptor * LAST_SEQNUM
• static const MamaFieldDescriptor * TOTAL_VOLUME_SEQNUM
• static const MamaFieldDescriptor * UNIQUE_ID
• static const MamaFieldDescriptor * TRADE_ACTION
• static const MamaFieldDescriptor * TRADE_EXEC_VENUE
• static const MamaFieldDescriptor * OFF_EXCHANGE_TRADE_PRICE
• static const MamaFieldDescriptor * ON_EXCHANGE_TRADE_PRICE
• static const MamaFieldDescriptor * TRADE_CONTRIBUTORS
• static const MamaFieldDescriptor * GENERIC_FLAG
• static const MamaFieldDescriptor * TRADE_RECAPS
• static const MamaFieldDescriptor * SHORT_SALE_CIRCUIT_BREAKER
• static const MamaFieldDescriptor * ORIG_SHORT_SALE_CIRCUIT_BREAKER
• static const MamaFieldDescriptor * CORR_SHORT_SALE_CIRCUIT_BREAKER

6.139.1 Detailed Description

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing trade related fields from update messages.

This class should be initialized prior to using the MamdaTradeListener by calling setDictionary() with a valid dictionary object which contains trade related fields.

6.139.2 Member Function Documentation

6.139.2.1 static void Wombat::MamdaTradeFields::setDictionary (const MamaDictionary & dictionary) [static]

6.139.2.2 static void Wombat::MamdaTradeFields::reset () [static]

Reset the dictionary for trade update fields.
6.139.2.3  static uint16_t Wombat::MamdaTradeFields::getMaxFid ()
          [static]

6.139.2.4  static bool Wombat::MamdaTradeFields::isSet ()  [static]
6.139.3  Member Data Documentation

6.139.3.1  const MamaFieldDescriptor* Wombat::MamdaTradeFields::TRADE_PRICE  [static]

6.139.3.2  const MamaFieldDescriptor* Wombat::MamdaTradeFields::TRADE_DATE  [static]

6.139.3.3  const MamaFieldDescriptor* Wombat::MamdaTradeFields::TRADE_SIDE  [static]

6.139.3.4  const MamaFieldDescriptor* Wombat::MamdaTradeFields::AGGRESSOR_SIDE  [static]

6.139.3.5  const MamaFieldDescriptor* Wombat::MamdaTradeFields::TRADE_TIME  [static]

6.139.3.6  const MamaFieldDescriptor* Wombat::MamdaTradeFields::LAST_PRICE  [static]

6.139.3.7  const MamaFieldDescriptor* Wombat::MamdaTradeFields::LAST_VOLUME  [static]

6.139.3.8  const MamaFieldDescriptor* Wombat::MamdaTradeFields::LAST_DATE_TIME  [static]

6.139.3.9  const MamaFieldDescriptor* Wombat::MamdaTradeFields::LAST_PART_ID  [static]

6.139.3.10 const MamaFieldDescriptor* Wombat::MamdaTradeFields::LAST_DIRECTION  [static]

6.139.3.11 const MamaFieldDescriptor* Wombat::MamdaTradeFields::NET_CHANGE  [static]

6.139.3.12 const MamaFieldDescriptor* Wombat::MamdaTradeFields::PCT_CHANGE  [static]

6.139.3.13 const MamaFieldDescriptor* Wombat::MamdaTradeFields::TRADE_SIZE  [static]

6.139.3.14 const MamaFieldDescriptor* Wombat::MamdaTradeFields::TOTAL_VOLUME  [static]

6.139.3.15 const MamaFieldDescriptor* Wombat::MamdaTradeFields::OFF_EXCHANGE_TOTAL_VOLUME
          [static]

6.139.3.16 const MamaFieldDescriptor* Wombat::MamdaTradeFields::ON_EXCHANGE_TOTAL_VOLUME
          [static]
- MamdaTradeFields.h
6.140 Wombat::MamdaTradeGap Class Reference

*MamdaTradeGap* is an interface that provides access to trade gap related fields.

```cpp
#include <MamdaTradeGap.h>
```

Inheritance diagram for Wombat::MamdaTradeGap:
```
Wombat::MamdaTradeGap
Wombat::MamdaBasicEvent
Wombat::MamdaTradeListener
```

Public Member Functions

- virtual `mama_seqnum_t getBeginGapSeqNum () const =0`
  
  The starting sequence number of detected missing trades based on the trade count.

- virtual `MamdaFieldState getBeginGapSeqNumFieldState () const =0`
- virtual `mama_seqnum_t getEndGapSeqNum () const =0`
  
  The end sequence number of detected missing trades based on the trade count.

- virtual `MamdaFieldState getEndGapSeqNumFieldState () const =0`
- virtual `~MamdaTradeGap ()`

6.140.1 Detailed Description

*MamdaTradeGap* is an interface that provides access to trade gap related fields.

6.140.2 Constructor & Destructor Documentation

6.140.2.1 virtual `Wombat::MamdaTradeGap::~MamdaTradeGap ()`

```cpp
68 {};
```
6.140.3 Member Function Documentation

6.140.3.1 virtual mama_seqnum_t Wombat::MamdaTradeGap::getBeginGapSeqNum () const [pure virtual]

The starting sequence number of detected missing trades based on the trade count.

Returns:

The start of the sequence number gap.

Implemented in Wombat::MamdaTradeListener.

6.140.3.2 virtual MamdaFieldState Wombat::MamdaTradeGap::getBeginGapSeqNumFieldState () const [pure virtual]

Returns:

The Field State of the start of the sequence number gap.

Implemented in Wombat::MamdaTradeListener.

6.140.3.3 virtual mama_seqnum_t Wombat::MamdaTradeGap::getEndGapSeqNum () const [pure virtual]

The end sequence number of detected missing trades based on the trade count.

Returns:

The end of the sequence number gap.

Implemented in Wombat::MamdaTradeListener.

6.140.3.4 virtual MamdaFieldState Wombat::MamdaTradeGap::getEndGapSeqNumFieldState () const [pure virtual]

Returns:

The field state of the end of the sequence number gap.

Implemented in Wombat::MamdaTradeListener.

The documentation for this class was generated from the following file:

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
• MamdaTradeGap.h
MamdaTradeHandler is an interface for applications that want to have an easy way to handle trade updates.

```cpp
#include <MamdaTradeHandler.h>
```

### Public Member Functions

- `virtual void onTradeRecap (MamdaSubscription *subscription, MamdaTradeListener &listener, const MamaMsg &msg, const MamdaTradeRecap &recap)=0`
  
  Method invoked when the current last-trade information for the security is available.

- `virtual void onTradeReport (MamdaSubscription *subscription, MamdaTradeListener &listener, const MamaMsg &msg, const MamdaTradeReport &event, const MamdaTradeRecap &recap)=0`
  
  Method invoked when a trade is reported.

- `virtual void onTradeGap (MamdaSubscription *subscription, MamdaTradeListener &listener, const MamaMsg &msg, const MamdaTradeGap &event, const MamdaTradeRecap &recap)=0`
  
  Method invoked when a gap in trade reports is discovered.

- `virtual void onTradeCancelOrError (MamdaSubscription *subscription, MamdaTradeListener &listener, const MamaMsg &msg, const MamdaTradeCancelOrError &event, const MamdaTradeRecap &recap)=0`
  
  Method invoked when a trade cancel or error is reported.

- `virtual void onTradeCorrection (MamdaSubscription *subscription, MamdaTradeListener &listener, const MamaMsg &msg, const MamdaTradeCorrection &event, const MamdaTradeRecap &recap)=0`
  
  Method invoked when a trade correction is reported.

- `virtual void onTradeClosing (MamdaSubscription *subscription, MamdaTradeListener &listener, const MamaMsg &msg, const MamdaTradeClosing &event, const MamdaTradeRecap &recap)=0`
  
  Method invoked for a closing report.

- `virtual void onTradeOutOfSequence (MamdaSubscription *subscription, MamdaTradeListener &listener, const MamaMsg &msg, const MamdaTradeOutOfSequence &event, const MamdaTradeRecap &recap)=0`
  
  Method invoked for a message marked as out of sequence.
• virtual void onTradePossiblyDuplicate (MamdaSubscription *subscription,
  MamdaTradeListener &listener, const MamaMsg &msg, const MamdaTrade-
  PossiblyDuplicate &event, const MamdaTradeRecap &recap)=0

  Method invoked for a message which is marked as possibly duplicate Listener must
  be configured to check the Msg Qualifier, i.e., call setControlProcessingBy-
  MsgQual on listener passing a value of true:

• virtual ~MamdaTradeHandler ()

6.141.1 Detailed Description

MamdaTradeHandler is an interface for applications that want to have an easy way to
handle trade updates.

The interface defines callback methods for different types of trade-related events:
trades, errors/cancels, corrections, recaps and closing reports.

6.141.2 Constructor & Destructor Documentation

6.141.2.1 virtual Wombat::MamdaTradeHandler::
  ~MamdaTradeHandler ()
  [virtual]

181 {};

6.141.3 Member Function Documentation

6.141.3.1 virtual void Wombat::MamdaTradeHandler::onTradeRecap
  (MamdaSubscription * subscription, MamdaTradeListener & listener,
  const MamaMsg & msg, const MamdaTradeRecap & recap)  [pure
  virtual]

Method invoked when the current last-trade information for the security is available.
The reason for the invocation may be any of the following:

• Initial image.
• Recap update (e.g., after server fault tolerant event or data quality event.)
• After stale status removed.

Parameters:

  subscription  The subscription which received the update.
  listener  The listener which invoked the callback.
6.141 Wombat::MamdaTradeHandler Class Reference

\[ msg \] The MamaMsg that triggered this invocation.

\[ recap \] Access to the details in the trade recap.

6.141.3.2 virtual void Wombat::MamdaTradeHandler::onTradeReport (MamdaSubscription * subscription, MamdaTradeListener & listener,
const MamaMsg & msg, const MamdaTradeReport & event, const
MamdaTradeRecap & recap) [pure virtual]

Method invoked when a trade is reported.

Parameters:

\[ subscription \] The subscription which received this update.

\[ listener \] The listener which invoked this callback.

\[ msg \] The MamaMsg that triggered this invocation.

\[ event \] Access to the trade report event details.

\[ recap \] Access to all trade related fields.

6.141.3.3 virtual void Wombat::MamdaTradeHandler::onTradeGap (MamdaSubscription * subscription, MamdaTradeListener & listener,
const MamaMsg & msg, const MamdaTradeGap & event, const
MamdaTradeRecap & recap) [pure virtual]

Method invoked when a gap in trade reports is discovered.

Parameters:

\[ subscription \] The subscription which detected the gap.

\[ listener \] The listener which invoked this callback.

\[ msg \] The MamaMsg which triggered this invocation.

\[ event \] Access to the gap event details.

\[ recap \] Access to the complete trade information.

6.141.3.4 virtual void Wombat::MamdaTradeHandler::onTradeCancelOr-
Error (MamdaSubscription * subscription, MamdaTradeListener &
listener, const MamaMsg & msg, const MamdaTradeCancelOrError
& event, const MamdaTradeRecap & recap) [pure virtual]

Method invoked when a trade cancel or error is reported.
Parameters:

- **subscription** The subscription which received the update.
- **listener** The listener which invoked this callback.
- **msg** The MamaMsg that triggered this invocation.
- **event** Access to the details of the trade cancel or error event.
- **recap** Access to the full trade details.

### 6.141.3.5 virtual void Wombat::MamdaTradeHandler::onTradeCorrection

```
(MamdaSubscription * subscription, MamdaTradeListener & listener,
 const MamaMsg & msg, const MamdaTradeCorrection & event, const
 MamdaTradeRecap & recap) [pure virtual]
```

Method invoked when a trade correction is reported.

Parameters:

- **subscription** The subscription which received the update.
- **listener** The listener which invoked this callback.
- **msg** The MamaMsg that triggered this invocation.
- **event** Access to the trade correction event details.
- **recap** Access to the full trade details.

### 6.141.3.6 virtual void Wombat::MamdaTradeHandler::onTradeClosing

```
(MamdaSubscription * subscription, MamdaTradeListener & listener,
 const MamaMsg & msg, const MamdaTradeClosing & event, const
 MamdaTradeRecap & recap) [pure virtual]
```

Method invoked for a closing report.

Parameters:

- **subscription** The subscription which received the update.
- **listener** The listener which invoked this callback.
- **msg** The MamaMsg that triggered this invocation.
- **event** Access to the trade closing event details.
- **recap** Access to the full trade details.
Method invoked for a message marked as out of sequence.

Listener must be configured to check the Msg Qualifier, i.e., call `setControlProcessingByMsgQual()` on listener passing a value of `true`;

**Parameters:**

- `subscription` The subscription which received the update.
- `listener` The trade listener which invoked this callback.
- `msg` The MamaMsg that triggered this invocation.
- `event` Details on the out of sequence trade event.
- `recap` Access to the full trade details.

Method invoked for a message which is marked as possibly duplicate Listener must be configured to check the Msg Qualifier, i.e., call `setControlProcessingByMsgQual()` on listener passing a value of `true`;

**Parameters:**

- `subscription` The subscription which received the update.
- `listener` The trade listener which invoked this callback.
- `msg` The MamaMsg that triggered this invocation.
- `event` Details on the possibly duplicate trade event.
- `recap` Access to the full trade details.

The documentation for this class was generated from the following file:

- `MamdaTradeHandler.h`
6.142 Wombat::MamdaTradeListener Class Reference

MamdaTradeListener is a class that specializes in handling trade updates.

#include <MamdaTradeListener.h>

Inheritance diagram for Wombat::MamdaTradeListener:

```
Wombat::MamdaMsgListener
  Wombat::MamdaTradeRecap
    Wombat::MamdaTradeReport
      Wombat::MamdaTradeGap
        Wombat::MamdaTradeCancelOrError
          Wombat::MamdaTradeCorrection
            Wombat::MamdaTradeClosing
              Wombat::MamdaTradeOutOfSequence
                Wombat::MamdaTradePossiblyDuplicate
          Wombat::MamdaTradeRecap
    Wombat::MamdaTradeReport
      Wombat::MamdaTradeGap
        Wombat::MamdaTradeCancelOrError
          Wombat::MamdaTradeCorrection
            Wombat::MamdaTradeClosing
              Wombat::MamdaTradeOutOfSequence
                Wombat::MamdaTradePossiblyDuplicate
          Wombat::MamdaTradeRecap
    Wombat::MamdaTradeReport
      Wombat::MamdaTradeGap
        Wombat::MamdaTradeCancelOrError
          Wombat::MamdaTradeCorrection
            Wombat::MamdaTradeClosing
              Wombat::MamdaTradeOutOfSequence
                Wombat::MamdaTradePossiblyDuplicate
          Wombat::MamdaTradeRecap
    Wombat::MamdaTradeReport
      Wombat::MamdaTradeGap
        Wombat::MamdaTradeCancelOrError
          Wombat::MamdaTradeCorrection
            Wombat::MamdaTradeClosing
              Wombat::MamdaTradeOutOfSequence
                Wombat::MamdaTradePossiblyDuplicate
          Wombat::MamdaTradeRecap
    Wombat::MamdaTradeReport
```

Public Member Functions

- MamdaTradeListener()
- virtual ~MamdaTradeListener()
- void addHandler(MamdaTradeHandler *handler)
- void processPosDupAndOutOfSeqAsTransient(bool tf)
- void resolvePossiblyDuplicate(bool tf)
- void usePosDupAndOutOfSeqHandlers(bool tf)
- void setCheckUpdatesForTrades(bool check)
- const char *getSide() const
- const char *getSymbol() const
Get the instruments string symbol.

- const char * getPartId () const
  Get the participant identifier.

- const MamaDateTime & getSrcTime () const
  Get the source time.

- const MamaDateTime & getActivityTime () const
  Get the activity time.

- const MamaDateTime & getLineTime () const
  Get the line time.

- const MamaDateTime & getSendTime () const
  Get the send time.

- const MamaMsgQual & getMsgQual () const
  Get the message qualifier.

- const char * getPubId () const

- mama_seqnum_t getEventSeqNum () const
  Sequence number of trade.

- const MamaDateTime & getEventTime () const
  Get the event time.

- const MamaPrice & getLastPrice () const
  Monetary value of an individual share of the security at the time of the trade.

- mama_quantity_t getLastVolume () const
  Number of shares traded in a single transaction for an individual security.

- const char * getLastPartId () const
  Trade participant ID.

- const MamaDateTime & getLastTime () const
  Time corresponding to the last trade, as reported by the feed.

- const MamaPrice & getIrregPrice () const
  Monetary value of an individual share of the security at the time of the last irregular trade.
• mama_quantity_t getIrregVolume () const
  Number of shares traded in a single transaction for an individual security.

• const char * getIrregPartId () const
  Irregular trade participant ID.

• const MamaDateTime & getIrregTime () const
  Time corresponding to the last irregular trade, as reported by the feed.

• const MamaDateTime & getTradeDate () const
  Time corresponding to the last trade, as reported by the feed.

• mama_u32_t getTradeCount () const
  The number of trades today.

• mama_quantity_t getAccVolume () const
  Total volume of shares traded in a security at the time it is disseminated.

• mama_quantity_t getOffExAccVolume () const
  Total volume of off-exchange shares traded in a security at the time it is disseminated.

• mama_quantity_t getOnExAccVolume () const
  Total volume of on-exchange shares traded in a security at the time it is disseminated.

• const MamaPrice & getNetChange () const
  Change in price compared with the previous closing price (i.e.

• double getPctChange () const
  Percentage change in price compared with the previous closing price (i.e.

• MamdaTradeDirection getTradeDirection () const
  Trade tick direction.

• const MamaPrice & getOpenPrice () const
  The price of the first qualifying trade in the security during the current trading day.

• const MamaPrice & getHighPrice () const
  Highest price paid for security during the trading day.

• const MamaPrice & getLowPrice () const
  Lowest price paid for security during the trading day.
• const MamaPrice & getClosePrice () const
  
  Return the Close price Today’s closing price.

• const MamaPrice & getPrevClosePrice () const
  
  The last qualifying trade price on the previous trading day.

• const MamaPrice & getAdjPrevClosePrice () const
  
  The previous close price adjusted by corporate actions, such as dividends and stock
  splits on the ex-date.

• const MamaDateTime & getPrevCloseDate () const
  
  Date corresponding to wPrevClosePrice.

• mama_u32_t getBlockCount () const
  
  The number of block trades (at least 10,000 shares) today.

• mama_quantity_t getBlockVolume () const
  
  Total volume of block trades today.

• double getVwap () const
  
  Volume-weighted average price of a security at the time it is disseminated.

• double getOffExVwap () const
  
  Volume-weighted average price of an off-exchange security at the time it is dissemi-
  nated.

• double getOnExVwap () const
  
  Volume-weighted average price of an on-exchange security at the time it is dissemi-
  nated.

• double getTotalValue () const
  
  Total value of all shares traded in a security at the time it is disseminated.

• double getOffExTotalValue () const
  
  Total value of all off-exchange shares traded in a security at the time it is dissemi-
  nated.

• double getOnExTotalValue () const
  
  Total value of all on-exchange shares traded in a security at the time it is disseminated.

• double getStdDev () const
  
  Standard deviation of last trade price of a security at the time it is disseminated.
• double getStdDevSum () const
  Sum of the standard deviations.

• double getStdDevSumSquares () const
  Square of the sum of the standard deviations.

• const char * getTradeUnits () const
  Reuters trade units.

• mama_seqnum_t getLastSeqNum () const
  Sequence number of the last trade.

• mama_seqnum_t getHighSeqNum () const
  Sequence number of incoming message which gives high value.

• mama_seqnum_t getLowSeqNum () const
  Sequence number of incoming message which gives low value.

• mama_seqnum_t getTotalVolumeSeqNum () const

• const char * getCurrencyCode () const
  Currency of the trade (eg US$).

• const MamaPrice & getSettlePrice () const
  Settle price of trade.

• const MamaDateTime & getSettleDate () const
  Settle date of trade.

• MamaTradeExecVenue getTradeExecVenue () const
  Trade execution venue.

• const MamaPrice & getOffExchangeTradePrice () const
  Monetary value of an individual share of the security off exchange at the time of the trade.

• const MamaPrice & getOnExchangeTradePrice () const
  Monetary value of an individual share of the security on exchange at the time of the trade.

• const MamaPrice & getTradePrice () const
  Get the trade price.

• mama_quantity_t getTradeVolume () const
Get the trade volume.

- const char * getTradePartId () const
  Get the participant identifier for the trade.

- const char * getTradeQual () const
  Get the wombat normalized trade qualifier.

- const char * getTradeQualNative () const
  Get the Trade condition ("sale condition").

- mama_u32_t getTradeSellersSaleDays () const
  Get the seller’s sale days.

- char getTradeStopStock () const
  Get the Stopped stock indicator.

- bool getIsIrregular () const
  Get whether the trade is irregular.

- mama_u64_t getOrderId () const
  Get the trade’s order id, if available.

- const char * getUniqueId () const
  Get the unique ID.

- const char * getTradeId () const
  Get the trade id.

- const char * getCorrTradeId () const
  Get the corrected trade Id.

- const char * getTradeAction () const
  Get the trade action.

- mama_seqnum_t getBeginGapSeqNum () const
  The starting sequence number of detected missing trades based on the trade count.

- mama_seqnum_t getEndGapSeqNum () const
  The end sequence number of detected missing trades based on the trade count.

- bool getIsCancel () const
Return whether this event is a trade cancel.

- mama_seqnum_t getOrigSeqNum () const
  
  Original feed-generated sequence for a correction/cancel/error.

- const MamaPrice & getOrigPrice () const
  
  Original trade price in a correction/cancel/error.

- mama_quantity_t getOrigVolume () const
  
  Original trade size in a correction/cancel/error.

- const char * getOrigPartId () const
  
  Original trade participant identifier in a correction/cancel/error.

- const char * getOrigQual () const
  
  A normalized set of qualifiers for the original trade for the security in a correc-
  tion/cancel/error.

- const char * getOrigQualNative () const
  
  Feed-specific trade qualifier code(s) for original trade.

- mama_u32_t getOrigSellersSaleDays () const
  
  Seller's sale days for original trade.

- char getOrigStopStock () const
  
  Stopped stock indicator for original trade.

- const char * getOrigTradeId () const
  
  Get the original trade id.

- bool getGenericFlag () const

- char getShortSaleCircuitBreaker () const
  
  get the ShortSaleCircuitBreaker

- char getOrigShortSaleCircuitBreaker () const
  
  get the OrigShortSaleCircuitBreaker

- char getCorrShortSaleCircuitBreaker () const
  
  get the CorrShortSaleCircuitBreaker

- const MamaPrice & getCorrPrice () const
  
  Get the corrected trade price.
• mama_quantity_t getCorrVolume () const
  Get the corrected trade volume.

• const char * getCorrPartId () const
  Get the corrected trade participant identifier.

• const char * getCorrQual () const
  Get corrected trade qualifier.

• const char * getCorrQualNative () const
  Get corrected trade condition.

• mama_u32_t getCorrSellersSaleDays () const
  Get the corrected trade sellers days.

• char getCorrStopStock () const
  Get the original stock stop indicator.

• bool getIsIndicative () const
  Return whether this closing price is indicative or official.

• MamdaFieldState getSymbolFieldState () const
  Get the string symbol field state.

• MamdaFieldState getPartIdFieldState () const
  Get the participant identifier field state.

• MamdaFieldState getSrcTimeFieldState () const
  Get the source time field state.

• MamdaFieldState getActivityTimeFieldState () const
  Get the activity time field state.

• MamdaFieldState getLineTimeFieldState () const
  Get the line time of the update.

• MamdaFieldState getSendTimeFieldState () const
  Get the send time field state.

• MamdaFieldState getMsgQualFieldState () const
  Get the message qualifier field state.
• MamdaFieldState getPubIdFieldState () const
  The event SeqNum Field State.

• MamdaFieldState getEventSeqNumFieldState () const
  Get the event time field state.

• MamdaFieldState getLastPriceFieldState () const
  The last trade price Field State.

• MamdaFieldState getLastVolumeFieldState () const
  The last volume Field State.

• MamdaFieldState getLastPartIdFieldState () const
  The last part Id Field State.

• MamdaFieldState getLastTimeFieldState () const
  The last time Field State.

• MamdaFieldState getIrregPriceFieldState () const
  The irreg price Field State.

• MamdaFieldState getIrregVolumeFieldState () const
  The irreg volume Field State.

• MamdaFieldState getIrregPartIdFieldState () const
  The irreg part Id Field State.

• MamdaFieldState getIrregTimeFieldState () const
  The irregular time Field State.

• MamdaFieldState getTradeDateFieldState () const
  The trade date Field State.

• MamdaFieldState getSideFieldState () const
  The TradeSide or AggressorSide Field State.

• MamdaFieldState getTradeCounFieldState () const
  MamdaFieldState getAccVolumeFieldState () const
  The accumulated volume Field State.

• MamdaFieldState getOffExAccVolumeFieldState () const
The off exchange accumulated volume Field State.

- `MamdaFieldState getOnExAccVolumeFieldState()` const
  The on exchange accumulated volume Field State.

- `MamdaFieldState getNetChangeFieldState()` const
  The net change Field State.

- `MamdaFieldState getPctChangeFieldState()` const
  The percentage change Field State.

- `MamdaFieldState getTradeDirectionFieldState()` const
  The trade direction Field State.

- `MamdaFieldState getOpenPriceFieldState()` const
  The open price Field State.

- `MamdaFieldState getHighPriceFieldState()` const
  The high price Field State.

- `MamdaFieldState getLowPriceFieldState()` const
  The low price Field State.

- `MamdaFieldState getClosePriceFieldState()` const
  Get the field state.

- `MamdaFieldState getPrevClosePriceFieldState()` const
  The previous close price Field State.

- `MamdaFieldState getAdjPrevClosePriceFieldState()` const
  The adjusted previous close date Field State.

- `MamdaFieldState getPrevCloseDateFieldState()` const
  The previous close date Field State.

- `MamdaFieldState getBlockCountFieldState()` const
  The block count Field State.

- `MamdaFieldState getBlockVolumeFieldState()` const
  The block volume Field State.

- `MamdaFieldState getVwapFieldState()` const
The vwap Field State.

- MamdaFieldState getOffExVwapFieldState() const
  The off exchange vwap Field State.

- MamdaFieldState getOnExVwapFieldState() const
  The on exchange vwap Field State.

- MamdaFieldState getTotalValueFieldState() const
  The total value Field State.

- MamdaFieldState getOffExTotalValueFieldState() const
  The Off Exchange Total Value Field State.

- MamdaFieldState getOnExTotalValueFieldState() const
  The On Exchange Total Value Field State.

- MamdaFieldState getStdDevFieldState() const
  The std deviation Field State.

- MamdaFieldState getStdDevSumFieldState() const
  The std deviation sum Field State.

- MamdaFieldState getStdDevSumSquaresFieldState() const
  The StdDevSumSquares Field State.

- MamdaFieldState getTradeUnitsFieldState() const
  The trade units Field State.

- MamdaFieldState getLastSeqNumFieldState() const
  The last SeqNum Field State.

- MamdaFieldState getHighSeqNumFieldState() const
  The high SeqNum Field State.

- MamdaFieldState getLowSeqNumFieldState() const
  The low SeqNum Field State.

- MamdaFieldState getTotalVolume SeqNumFieldState() const
  The total volume seqNum Field State.

- MamdaFieldState getCurrencyCodeFieldState() const
The currency code Field State.

- `MamdaFieldState getSettlePriceFieldState () const`
  The settle price Field State.

- `MamdaFieldState getSettleDateFieldState () const`
  The last trade price Field State.

- `MamdaFieldState getTradeExecVenuFieldState () const`
- `MamdaFieldState getOffExchangeTradePriceFieldState () const`
  The settle date Field State.

- `MamdaFieldState getOnExchangeTradePriceFieldState () const`
  The onExchange trade price Field State.

- `MamdaFieldState getTradePriceFieldState () const`
  Get the field state.

- `MamdaFieldState getTradeVolumeFieldState () const`
  Get the field state.

- `MamdaFieldState getTradePartIdFieldState () const`
  Get the field state.

- `MamdaFieldState getTradeQualFieldState () const`
  The trade qual Field State.

- `MamdaFieldState getTradeQualNativeFieldState () const`
  Get the field state.

- `MamdaFieldState getTradeSellersSaleDaysFieldState () const`
  Get the field state.

- `MamdaFieldState getTradeStopStockFieldState () const`
  Get the field state.

- `MamdaFieldState getIsIrregularFieldState () const`
  The isIrregular Field State.

- `MamdaFieldState getOrderIdFieldState () const`
  Get the field state.

- `MamdaFieldState getUniqueIdFieldState () const`
The unique ID Field State.

- `MamdaFieldState getTradeIdFieldState () const`  
  The trade ID Field State.

- `MamdaFieldState getCorrTradeIdFieldState () const`  
  Get the field state.

- `MamdaFieldState getTradeActionFieldState () const`  
  The trade action Field State.

- `MamdaFieldState getTradeExecVenueFieldState () const`  
- `MamdaFieldState getBeginGapSeqNumFieldState () const`  
- `MamdaFieldState getEndGapSeqNumFieldState () const`  
- `MamdaFieldState getIsCancelFieldState () const`  
- `MamdaFieldState getOrigSeqNumFieldState () const`  
  Get the field state.

- `MamdaFieldState getOrigPriceFieldState () const`  
  Get the field state.

- `MamdaFieldState getOrigVolumeFieldState () const`  
  Get the field state.

- `MamdaFieldState getOrigPartIdFieldState () const`  
  Get the field state.

- `MamdaFieldState getOrigQualFieldState () const`  
  Get the field state.

- `MamdaFieldState getOrigQualNativeFieldState () const`  
  Get the field state.

- `MamdaFieldState getOrigSellersSaleDaysFieldState () const`  
  Get the field state.

- `MamdaFieldState getOrigStopStockFieldState () const`  
  Get the field state.

- `MamdaFieldState getCorrPriceFieldState () const`  
  Get the field state.

- `MamdaFieldState getCorrVolumeFieldState () const`
Get the field state.

- MamdaFieldState getCorrPartIdFieldState () const
  Get the field state.

- MamdaFieldState getCorrQualFieldState () const
  Get the field state.

- MamdaFieldState getCorrQualNativeFieldState () const
  Get the field state.

- MamdaFieldState getCorrSellersSaleDaysFieldState () const
  Get the field state.

- MamdaFieldState getCorrStopStockFieldState () const
  Get the field state.

- MamdaFieldState getIsIndicativeFieldState () const
- MamdaFieldState getTradeCountFieldState () const
  The trade count Field State.

- MamdaFieldState getOrigTradeIdFieldState () const
- MamdaFieldState getGenericFlagFieldState () const
- MamdaFieldState getShortSaleCircuitBreakerFieldState () const
- MamdaFieldState getOrigShortSaleCircuitBreakerFieldState () const
- MamdaFieldState getCorrShortSaleCircuitBreakerFieldState () const
  virtual void onMsg (MamdaSubscription *subscription, const MamaMsg &msg, short msgType)
  Implementation of MamdaListener interface.

- void assertEqual (MamdaTradeListener *listener)
- void reset (void)

6.142.1 Detailed Description

MamdaTradeListener is a class that specializes in handling trade updates.

Developers provide their own implementation of the MamdaTradeHandler interface and will be delivered notifications for trades, trade cancels/error/corrections, and trade closing prices. An obvious application for this MAMDA class is any kind of trade tick capture application.
Note: The `MamdaTradeListener` class caches trade-related field values. Among other reasons, caching of these fields makes it possible to provide complete trade-related callbacks, even when the publisher (e.g., feed handler) is only publishing deltas containing modified fields.

For details on the accessor methods for the cache data see the description for the corresponding methods on the specific MamdaEvent derived classes.

`MamdaTradeListener` should initialize the `MamdaTradeFields` class prior to receiving the first message by calling `MamdaTradeFields::setDictionary()` with a valid dictionary object which contains Trade related fields.

### 6.142.2 Constructor & Destructor Documentation

#### 6.142.2.1 `Wombat::MamdaTradeListener::MamdaTradeListener ()`

#### 6.142.2.2 `virtual Wombat::MamdaTradeListener::~MamdaTradeListener ()`

[virtual]

### 6.142.3 Member Function Documentation

#### 6.142.3.1 `void Wombat::MamdaTradeListener::addHandler (MamdaTradeHandler * handler)`

#### 6.142.3.2 `void Wombat::MamdaTradeListener::processPosDupAndOutOfSeqAsTransient (bool tf)`

#### 6.142.3.3 `void Wombat::MamdaTradeListener::resolvePossiblyDuplicate (bool tf)`

#### 6.142.3.4 `void Wombat::MamdaTradeListener::usePosDupAndOutOfSeqHandlers (bool tf)`

#### 6.142.3.5 `void Wombat::MamdaTradeListener::setCheckUpdatesForTrades (bool check)`

#### 6.142.3.6 `const char* Wombat::MamdaTradeListener::getSide () const`

Returns:

the Aggressor Side or TradeSide TradeSide
- 0 : No TradeSide is currently known/available.
- 1 or B : Buy
- 2 or S : Sell

Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
– 0 : No AggressorSide is currently known/available.
– 1 or B : Buy
– 2 or S : Sell

Implements Wombat::MamdaTradeRecap.

### 6.142.3.7 const char ∗ Wombat::MamdaTradeListener::getSymbol () const

[virtual]

Get the instruments string symbol.

**Returns:**

Symbol. This is the "well-known" symbol for the security, including any symbology mapping performed by the publisher.

Implements Wombat::MamdaBasicEvent.

### 6.142.3.8 const char ∗ Wombat::MamdaTradeListener::getPartId () const

[virtual]

Get the participant identifier.

**Returns:**

Participant ID. This may be an exchange identifier, a market maker ID, etc., or NULL (if this is not related to any specific participant).

Implements Wombat::MamdaBasicEvent.

### 6.142.3.9 const MamaDateTime& Wombat::MamdaTradeListener::getSrcTime () const

[virtual]

Get the source time.

**Returns:**

Source time. Typically, the exchange generated feed time stamp. This is often the same as the "event time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.
6.142.3.10 const MamaDateTime& Wombat::MamdaTradeListener::getActivityTime () const  [virtual]

Get the activity time.

Returns:

Activity time. A feed handler generated time stamp representing when the data item was last updated.

Implements Wombat::MamdaBasicEvent.

6.142.3.11 const MamaDateTime& Wombat::MamdaTradeListener::getLineTime () const  [virtual]

Get the line time.

Returns:

Line time. A feed handler (or similar publisher) time stamp representing the time that such publisher received the update message pertaining to the event. If clocks are properly synchronized and the source time (see above) is accurate enough, then the difference between the source time and line time is the latency between the data source and the feed handler.

Implements Wombat::MamdaBasicEvent.

6.142.3.12 const MamaDateTime& Wombat::MamdaTradeListener::getSendTime () const  [virtual]

Get the send time.

Returns:

Send time. A feed handler (or similar publisher) time stamp representing the time that such publisher sent the current message. The difference between the line time and send time is the latency within the feed handler itself. Also, if clocks are properly synchronized then the difference between the send time and current time is the latency within the market data distribution framework (i.e. MAMA and the underlying middleware). See MAMA API: MamaDateTime::currentTime()).

Implements Wombat::MamdaBasicEvent.
6.142 Wombat::MamdaTradeListener Class Reference

6.142.3.13 const MamaMsgQual& Wombat::MamdaTradeListener::getMsgQual () const [virtual]

Get the message qualifier.

Returns:
Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.142.3.14 const char∗ Wombat::MamdaTradeListener::getPubId () const

6.142.3.15 mama_seqnum_t Wombat::MamdaTradeListener::getEventSeqNum () const [virtual]

Sequence number of trade.

Returns:
Sequence number of trade.

Implements Wombat::MamdaTradeRecap.

6.142.3.16 const MamaDateTime& Wombat::MamdaTradeListener::getEventTime () const [virtual]

Get the event time.

Returns:
Event time. Typically, when the event actually occurred. This is often the same as the "source time", because many feeds do not distinguish between the actual event time and when the exchange sent the message.

Implements Wombat::MamdaBasicEvent.

6.142.3.17 const MamaPrice& Wombat::MamdaTradeListener::getLastPrice () const [virtual]

Monetary value of an individual share of the security at the time of the trade.

Returns:
The last trade price.
Implements Wombat::MamdaTradeRecap.

### 6.142.3.18 mama_quantity_t Wombat::MamdaTradeListener::getLastVolume () const [virtual]

Number of shares traded in a single transaction for an individual security.

**Returns:**

The last trade volume.

Implements Wombat::MamdaTradeRecap.

### 6.142.3.19 const char∗ Wombat::MamdaTradeListener::getLastPartId () const [virtual]

Trade participant ID.

This is typically an exchange ID, sometimes a market maker ID.

**Returns:**

The last trade participant identifier.

Implements Wombat::MamdaTradeRecap.

### 6.142.3.20 const MamaDateTime& Wombat::MamdaTradeListener::getLastTime () const [virtual]

Time corresponding to the last trade, as reported by the feed.

The exact time of the trade may not be available, since rules governing trade reporting allow for a trades to be reported within a specified time limit.

**Returns:**

The last trade time.

Implements Wombat::MamdaTradeRecap.

### 6.142.3.21 const MamaPrice& Wombat::MamdaTradeListener::getIrregPrice () const [virtual]

Monetary value of an individual share of the security at the time of the last irregular trade.
Returns:

The last irregular trade price.

Implements Wombat::MamdaTradeRecap.

6.142.3.22 mama_quantity_t Wombat::MamdaTradeListener::getIrregVolume () const [virtual]

Number of shares traded in a single transaction for an individual security.

Returns:

The last irregular trade volume.

Implements Wombat::MamdaTradeRecap.

6.142.3.23 const char∗ Wombat::MamdaTradeListener::getIrregPartId () const [virtual]

Irregular trade participant ID.
This is typically an exchange ID, sometimes a market maker ID.

Returns:

The last irregular trade participant identifier.

Implements Wombat::MamdaTradeRecap.

6.142.3.24 const MamaDateTime& Wombat::MamdaTradeListener::getIrregTime () const [virtual]

Time corresponding to the last irregular trade, as reported by the feed.
The exact time of the trade may not be available, since rules governing trade reporting allow for a trades to be reported within a specified time limit.

Returns:

The last irregular trade time.

Implements Wombat::MamdaTradeRecap.
6.142.3.25 const MamaDateTime& Wombat::MamdaTradeListener::getTradeDate () const [virtual]

Time corresponding to the last trade, as reported by the feed.
The exact time of the trade may not be available, since rules governing trade reporting allow for trades to be reported within a specified time limit.

Returns:
The last trade time.

Implements Wombat::MamdaTradeRecap.

6.142.3.26 mama_u32_t Wombat::MamdaTradeListener::getTradeCount () const [virtual]

The number of trades today.

Returns:
The number of trades so far today.

Implements Wombat::MamdaTradeRecap.

6.142.3.27 mama_quantity_t Wombat::MamdaTradeListener::getAccVolume () const [virtual]

Total volume of shares traded in a security at the time it is disseminated.

Returns:
Accumulated trade volume.

Implements Wombat::MamdaTradeRecap.

6.142.3.28 mama_quantity_t Wombat::MamdaTradeListener::getOffExAccVolume () const [virtual]

Total volume of off-exchange shares traded in a security at the time it is disseminated.

Returns:
Accumulated off-exchange trade volume.

Implements Wombat::MamdaTradeRecap.
6.142.3.29 mama_quantity_t Wombat::MamdaTradeListener::getOnExAccVolume () const [virtual]

Total volume of on-exchange shares traded in a security at the time it is disseminated.

Returns:

Accumulated on-exchange trade volume.

Implements Wombat::MamdaTradeRecap.

6.142.3.30 const MamaPrice& Wombat::MamdaTradeListener::getNetChange () const [virtual]

Change in price compared with the previous closing price (i.e. previous closing price - trade price).

Returns:

Price change compared to previous price.

Implements Wombat::MamdaTradeRecap.

6.142.3.31 double Wombat::MamdaTradeListener::getPctChange () const [virtual]

Percentage change in price compared with the previous closing price (i.e. previous closing price - trade price).

Returns:

Percentage price change compared to previous price.

Implements Wombat::MamdaTradeRecap.

6.142.3.32 MamdaTradeDirection Wombat::MamdaTradeListener::getTradeDirection () const [virtual]

Trade tick direction.

See MamdaTradeDirection.h for details.

Returns:

The tick direction.

Implements Wombat::MamdaTradeRecap.
6.142.3.33  const MamaPrice& Wombat::MamdaTradeListener::getOpenPrice () const [virtual]

The price of the first qualifying trade in the security during the current trading day.

Returns:

  The opening price.

Implements Wombat::MamdaTradeRecap.

6.142.3.34  const MamaPrice& Wombat::MamdaTradeListener::getHighPrice () const [virtual]

Highest price paid for security during the trading day.

Returns:

  The highest trade price for the day.

Implements Wombat::MamdaTradeRecap.

6.142.3.35  const MamaPrice& Wombat::MamdaTradeListener::getLowPrice ()
const [virtual]

Lowest price paid for security during the trading day.

Returns:

  The lowest trade price for the day.

Implements Wombat::MamdaTradeRecap.

6.142.3.36  const MamaPrice& Wombat::MamdaTradeListener::getClosePrice () const [virtual]

Return the Close price Today's closing price.

The close price is populated when official closing prices are sent by the feed after the session close.

Returns:

  The trade closing price.

Implements Wombat::MamdaTradeClosing.
6.142.3.37 const MamaPrice& Wombat::MamdaTradeListener::getPrevClosePrice () const [virtual]

The last qualifying trade price on the previous trading day.

This field may be copied from the close price field during the morning "roll" of records in the feedhandler, or it may be obtained from a secondary source, or it may be explicitly sent by the feed prior to the opening of trading for the current day.

**Returns:**

The last qualifying trade price on the previous trading day.

Implements Wombat::MamdaTradeRecap.

6.142.3.38 const MamaPrice& Wombat::MamdaTradeListener::getAdjPrevClosePrice () const [virtual]

The previous close price adjusted by corporate actions, such as dividends and stock splits on the ex-date.

**Returns:**

The adjusted previous closing price.

**See also:**

getPrevClosePrice()

Implements Wombat::MamdaTradeRecap.

6.142.3.39 const MamaDateTime& Wombat::MamdaTradeListener::getPrevCloseDate () const [virtual]

Date corresponding to wPrevClosePrice.

**Returns:**

The closing price from the previous trading day.

**See also:**

getPrevClosePrice().

Implements Wombat::MamdaTradeRecap.
6.142.3.40  mama_u32_t Wombat::MamdaTradeListener::getBlockCount () const [virtual]

The number of block trades (at least 10,000 shares) today.

Returns:

   Number of block trades.

Implements Wombat::MamdaTradeRecap.

6.142.3.41  mama_quantity_t Wombat::MamdaTradeListener::getBlockVolume () const [virtual]

Total volume of block trades today.

Returns:

   Total volume of block trades.

Implements Wombat::MamdaTradeRecap.

6.142.3.42  double Wombat::MamdaTradeListener::getVwap () const [virtual]

Volume-weighted average price of a security at the time it is disseminated.
Equivalent to dividing total value by total volume.

Returns:

   The VWAP (Volume-weighted average price)

Implements Wombat::MamdaTradeRecap.

6.142.3.43  double Wombat::MamdaTradeListener::getOffExVwap () const [virtual]

Volume-weighted average price of an off-exchange security at the time it is disseminated.
Equivalent to dividing the off-exchange total value by the off-exchange total volume.

Returns:

   The off-exchange VWAP (Volume-weighted average price)

Implements Wombat::MamdaTradeRecap.
### 6.142.3.44 double Wombat::MamdaTradeListener::getOnExVwap () const

```
[virtual]
```

Volume-weighted average price of an on-exchange security at the time it is disseminated.
Equivalent to dividing on-exchange total value by the on-exchange total volume.

**Returns:**

The on-exchange VWAP (Volume-weighted average price)

Implements Wombat::MamdaTradeRecap.

### 6.142.3.45 double Wombat::MamdaTradeListener::getTotalValue () const

```
[virtual]
```

Total value of all shares traded in a security at the time it is disseminated.
Calculated by summing the result of multiplying the trade price by trade volume for each qualifying trade.

**Returns:**

Total value of all instruments traded.

Implements Wombat::MamdaTradeRecap.

### 6.142.3.46 double Wombat::MamdaTradeListener::getOffExTotalValue () const

```
[virtual]
```

Total value of all off-exchange shares traded in a security at the time it is disseminated.
Calculated by summing the result of multiplying the trade price by trade volume for each qualifying trade.

**Returns:**

Total value of all off-exchange instruments traded.

Implements Wombat::MamdaTradeRecap.

### 6.142.3.47 double Wombat::MamdaTradeListener::getOnExTotalValue () const

```
[virtual]
```

Total value of all on-exchange shares traded in a security at the time it is disseminated.
Calculated by summing the result of multiplying the trade price by trade volume for each qualifying trade.

**Returns:**

Total value of all on-exchange instruments traded.

Implements `Wombat::MamdaTradeRecap`.

### 6.142.3.48 double Wombat::MamdaTradeListener::getStdDev () const 

[virtual]

Standard deviation of last trade price of a security at the time it is disseminated.

**Returns:**

The standard deviation of last trade price.

Implements `Wombat::MamdaTradeRecap`.

### 6.142.3.49 double Wombat::MamdaTradeListener::getStdDevSum () const 

[virtual]

Sum of the standard deviations.

**Returns:**

The sum of the standard deviations.

Implements `Wombat::MamdaTradeRecap`.

### 6.142.3.50 double Wombat::MamdaTradeListener::getStdDevSumSquares () const 

[virtual]

Square of the sum of the standard deviations.

**Returns:**

The square of the sum of the standard deviations.

Implements `Wombat::MamdaTradeRecap`.
6.142 Wombat::MamdaTradeListener Class Reference

6.142.3.51 const char* Wombat::MamdaTradeListener::getTradeUnits () const [virtual]

Returns:
Reuters trade units.

Implements Wombat::MamdaTradeRecap.

6.142.3.52 mama_seqnum_t Wombat::MamdaTradeListener::getLastSeqNum () const [virtual]

Returns:
The sequence number of the last trade.

Implements Wombat::MamdaTradeRecap.

6.142.3.53 mama_seqnum_t Wombat::MamdaTradeListener::getHighSeqNum () const [virtual]

Returns:
Sequence number of incoming message which gives high value.

Implements Wombat::MamdaTradeRecap.

6.142.3.54 mama_seqnum_t Wombat::MamdaTradeListener::getLowSeqNum () const [virtual]

Returns:
Sequence number of incoming message which gives low value.

Implements Wombat::MamdaTradeRecap.
6.142.3.55 mama_seqnum_t Wombat::MamdaTradeListener::getTotalVolumeSeqNum () const  [virtual]

Implements Wombat::MamdaTradeRecap.

6.142.3.56 const char∗ Wombat::MamdaTradeListener::getCurrencyCode () const  [virtual]

Currency of the trade (eg US$).

Returns:

Currency of the trade.

Implements Wombat::MamdaTradeRecap.

6.142.3.57 const MamaPrice& Wombat::MamdaTradeListener::getSettlePrice () const  [virtual]

Settle price of trade.

Returns:

Settle price of trade.

Implements Wombat::MamdaTradeRecap.

6.142.3.58 const MamaDateTime& Wombat::MamdaTradeListener::getSettleDate () const  [virtual]

Settle date of trade.

Returns:

Settle date of trade.

Implements Wombat::MamdaTradeRecap.

6.142.3.59 MamdaTradeExecVenue Wombat::MamdaTradeListener::getTradeExecVenue () const  [virtual]

Trade execution venue.

See MamdaTradeExecVenue.h for details.
Returns:

The trade execution venue.

Implements \texttt{Wombat::MamdaTradeRecap}.

\texttt{6.142.3.60 const MamaPrice& Wombat::MamdaTradeListener::getOffExchangeTradePrice () const [virtual]}

Monetary value of an individual share of the security off exchange at the time of the trade.

Returns:

The last off exchange trade price.

Implements \texttt{Wombat::MamdaTradeRecap}.

\texttt{6.142.3.61 const MamaPrice& Wombat::MamdaTradeListener::getOnExchangeTradePrice () const [virtual]}

Monetary value of an individual share of the security on exchange at the time of the trade.

Returns:

The last on exchange trade price.

Implements \texttt{Wombat::MamdaTradeRecap}.

\texttt{6.142.3.62 const MamaPrice& Wombat::MamdaTradeListener::getTradePrice () const [virtual]}

Get the trade price.

Returns:

The monetary value of an individual share of the security at the time of the trade.

Implements \texttt{Wombat::MamdaTradeOutOfSequence}. 

\hspace*{1.6cm} --

\hspace{1.6cm} Generated on Thu Feb 7 17:04:59 2013 for MAMDA C++ API by Doxygen
Get the trade volume.

Returns:
The number of shares traded in a single transaction for an individual security.

Implements Wombat::MamdaTradeOutOfSequence.

Get the participant identifier for the trade.
This is typically an exchange ID or a market maker ID.

Returns:
Trade participant ID.

Implements Wombat::MamdaTradeOutOfSequence.

Get the wombat normalized trade qualifier.

Returns:
Trade qualifier. A normalized set of qualifiers for the current trade for the security. This field may contain multiple string values, separated by the colon(\:) character.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Regular trade. A trade made without stated conditions is deemed regular way for settlement on the third + business day following the transaction + date.</td>
</tr>
<tr>
<td>Acquisition</td>
<td>A transaction made on the Exchange as a result of an Exchange acquisition.</td>
</tr>
<tr>
<td>Bunched</td>
<td>A trade representing an aggregate of two or more regular trades in a security occurring at the same price either simultaneously or within the same 60 second period, with no individual trade exceeding 10,000 shares.</td>
</tr>
<tr>
<td>Cash</td>
<td>A transaction which calls for the delivery of securities and payment on the same day the trade takes place.</td>
</tr>
<tr>
<td>Distribution</td>
<td>Sale of a large block of stock in such a manner that the price is not adversely affected.</td>
</tr>
<tr>
<td>BunchedSold</td>
<td>A bunched trade which is reported late</td>
</tr>
<tr>
<td>Rule155</td>
<td>To qualify as a 155 print, a specialist arranges for the sale of the block at one &quot;clean-up&quot; price or at the different price limits on his book. If the block is sold at a &quot;clean-up&quot; price, the specialist should execute at the same price all the executable buy orders on his book. The sale qualifier is only applicable for AMEX trades.</td>
</tr>
<tr>
<td>SoldLast</td>
<td>Sold Last is used when a trade prints in sequence but is reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>NextDay</td>
<td>A transaction which calls for delivery of securities on the first business day after the trade date.</td>
</tr>
<tr>
<td>Opened</td>
<td>Indicates an opening transaction that is printed out of sequence or reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>PriorRef</td>
<td>An executed trade that relates to an obligation to trade at an earlier point in the trading day or that refer to a prior reference price. This may be the result of an order that was lost or misplaced or a SelectNet order that was not executed on a timely basis.</td>
</tr>
<tr>
<td>SplitTrade</td>
<td>An execution in two markets when the specialist or Market Maker in the market first receiving the order across</td>
</tr>
</tbody>
</table>
Implements Wombat::MamdaTradeCancelOrError.

6.142.3.66  const char* Wombat::MamdaTradeListener::getTradeQualNative () const [virtual]

Get the Trade condition ("sale condition").
Feed-specific trade qualifier code(s). This field is provided primarily for completeness and/or troubleshooting.

Returns:
The Trade Condition.

See also:
getTradeQual()

Implements Wombat::MamdaTradeOutOfSequence.

6.142.3.67  mama_u32_t Wombat::MamdaTradeListener::getTradeSellersSaleDays () const [virtual]

Get the seller’s sale days.
Used when the trade qualifier is "Seller". Specifies the number of days that may elapse before delivery of the security.

Returns:
The Seller’s sale days.

Implements Wombat::MamdaTradeOutOfSequence.

6.142.3.68  char Wombat::MamdaTradeListener::getTradeStopStock () const [virtual]

Get the Stopped stock indicator.
Condition related to certain NYSE trading rules. This is not related to a halted security status. (0 == N/A; 1 == Applicable)

Returns:
Stopped stock indicator.

Implements Wombat::MamdaTradeOutOfSequence.
6.142.3.69 bool Wombat::MamdaTradeListener::getIsIrregular () const
[virtual]

Get whether the trade is irregular.

**Returns:**
Whether or not the trade qualifies as an irregular trade. In general, only "regular" trades qualify to update the official last price and high/low prices.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.70 mama_u64_t Wombat::MamdaTradeListener::getOrderId () const
[virtual]

Get the trade’s order id, if available.

**Returns:**
The trade message unique order id number (if available).

Implements Wombat::MamdaTradeOutOfSequence.

6.142.3.71 const char ∗ Wombat::MamdaTradeListener::getUniqueId () const
[virtual]

Get the unique ID.

**Returns:**
The unique ID

Implements Wombat::MamdaTradeReport.

6.142.3.72 const char ∗ Wombat::MamdaTradeListener::getTradeId () const
[virtual]

Get the trade id.

**Returns:**

the trade id

Implements Wombat::MamdaTradeReport.
6.142.3.73 const char* Wombat::MamdaTradeListener::getCorrTradeId () const [virtual]

Get the corrected trade Id.
Implements Wombat::MamdaTradeCorrection.

6.142.3.74 const char* Wombat::MamdaTradeListener::getTradeAction () const [virtual]

Get the trade action.

Returns:
   The trade action

Implements Wombat::MamdaTradeReport.

6.142.3.75 mama_seqnum_t Wombat::MamdaTradeListener::getBeginGapSeqNum () const [virtual]

The starting sequence number of detected missing trades based on the trade count.

Returns:
   The start of the sequence number gap.

Implements Wombat::MamdaTradeGap.

6.142.3.76 mama_seqnum_t Wombat::MamdaTradeListener::getEndGapSeqNum () const [virtual]

The end sequence number of detected missing trades based on the trade count.

Returns:
   The end of the sequence number gap.

Implements Wombat::MamdaTradeGap.

6.142.3.77 bool Wombat::MamdaTradeListener::getIsCancel () const [virtual]

Return whether this event is a trade cancel.
If false, the event is a trade error.
Returns:

Whether this is a trade cancel.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.78 mama_seqnum_t Wombat::MamdaTradeListener::getOrigSeqNum () const [virtual]

Original feed-generated sequence for a correction/cancel/error.

Returns:

The original sequence number.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.79 const MamaPrice& Wombat::MamdaTradeListener::getOrigPrice () const [virtual]

Original trade price in a correction/cancel/error.

Returns:

The original trade price.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.80 mama_quantity_t Wombat::MamdaTradeListener::getOrigVolume () const [virtual]

Original trade size in a correction/cancel/error.

Returns:

The original trade volume.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.81 const char* Wombat::MamdaTradeListener::getOrigPartId () const [virtual]

Original trade participant identifier in a correction/cancel/error.
Returns:

The original trade participant identifier.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.82  

const char* Wombat::MamdaTradeListener::getOrigQual () const  

[virtual]

A normalized set of qualifiers for the original trade for the security in a correction/cancel/error.

Returns:

The original trade qualifier.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.83  

const char* Wombat::MamdaTradeListener::getOrigQualNative ()  

const  

[virtual]

Feed-specific trade qualifier code(s) for original trade.
This field is provided primarily for completeness and/or troubleshooting.

Returns:

The original trade condition.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.84  

mama_u32_t Wombat::MamdaTradeListener::getOrigSellersSaleDays () const  

[virtual]

Seller’s sale days for original trade.
Used when the trade qualifier is "Seller". Specifies the number of days that may elapse before delivery of the security.

Returns:

The original seller’s sale days.

Implements Wombat::MamdaTradeCancelOrError.
6.142.3.85 char Wombat::MamdaTradeListener::getOrigStopStock () const
    [virtual]

Stopped stock indicator for original trade.
Condition related to certain NYSE trading rules. This is not related to a halted security status. (0 == N/A; 1 == Applicable)

Returns:
    The original stopped stock indicator.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.86 const char * Wombat::MamdaTradeListener::getOrigTradeId () const
    [virtual]

Get the original trade id.

Returns:
    the original trade id

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.87 bool Wombat::MamdaTradeListener::getGenericFlag () const
    [virtual]

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.88 char Wombat::MamdaTradeListener::getShortSaleCircuitBreaker () const
    [virtual]

get the ShortSaleCircuitBreaker

Returns:
    ShortSaleCircuitBreaker
    • return values:
    • Blank: Short Sale Restriction Not in Effect.
    • A: Short Sale Restriction Activated.
    • C: Short Sale Restriction Continued.
    • D: Sale Restriction Deactivated.
    • E: Sale Restriction in Effect.

Implements Wombat::MamdaTradeRecap.
6.142.3.89 char Wombat::MamdaTradeListener::getOrigShortSaleCircuitBreaker () const [virtual]

get the OrigShortSaleCircuitBreaker

**Returns:**

OrigShortSaleCircuitBreaker

- return values:
  - Blank: Short Sale Restriction Not in Effect.
  - A: Short Sale Restriction Activated.
  - C: Short Sale Restriction Continued.
  - D: Sale Restriction Deactivated.
  - E: Sale Restriction in Effect.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.90 char Wombat::MamdaTradeListener::getCorrShortSaleCircuitBreaker () const [virtual]

get the CorrShortSaleCircuitBreaker

**Returns:**

CorrShortSaleCircuitBreaker

- return values:
  - Blank: Short Sale Restriction Not in Effect.
  - A: Short Sale Restriction Activated.
  - C: Short Sale Restriction Continued.
  - D: Sale Restriction Deactivated.
  - E: Sale Restriction in Effect.

Implements Wombat::MamdaTradeCorrection.

6.142.3.91 const MamaPrice& Wombat::MamdaTradeListener::getCorrPrice () const [virtual]

Get the corrected trade price.

**See also:**

MamdaTradeReport::getTradePrice()

Implements Wombat::MamdaTradeCorrection.
6.142.3.92 mama_quantity_t Wombat::MamdaTradeListener::getCorrVolume () const [virtual]

Get the corrected trade volume.

See also:

MamdaTradeReport::getTradeVolume()

Implements Wombat::MamdaTradeCorrection.

6.142.3.93 const char∗ Wombat::MamdaTradeListener::getCorrPartId () const [virtual]

Get the corrected trade participant identifier.

See also:

MamdaTradeReport::getTradePartId()

Implements Wombat::MamdaTradeCorrection.

6.142.3.94 const char∗ Wombat::MamdaTradeListener::getCorrQual () const [virtual]

Get corrected trade qualifier.

See also:

MamdaTradeReport::getTradeQual()

Implements Wombat::MamdaTradeCorrection.

6.142.3.95 const char∗ Wombat::MamdaTradeListener::getCorrQualNative ()
const [virtual]

Get corrected trade condition.

See also:

MamdaTradeReport::getTradeCondition()

Implements Wombat::MamdaTradeCorrection.
6.142.3.96 mama_u32_t Wombat::MamdaTradeListener::getCorrSellersSaleDays () const [virtual]

Get the corrected trade sellers days.

See also:
MamdaTradeReport::getTradeSellersSaleDays()

Implements Wombat::MamdaTradeCorrection.

6.142.3.97 char Wombat::MamdaTradeListener::getCorrStopStock () const [virtual]

Get the original stock stop indicator.

See also:
MamdaTradeReport::getTradeStopStock()

Implements Wombat::MamdaTradeCorrection.

6.142.3.98 bool Wombat::MamdaTradeListener::getIsIndicative () const [virtual]

Return whether this closing price is indicative or official.

Returns:
Whether the closing price is indicative or otherwise.

Implements Wombat::MamdaTradeClosing.

6.142.3.99 MamdaFieldState Wombat::MamdaTradeListener::getSymbolFieldState () const [virtual]

Get the string symbol field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.142 Wombat::MamdaTradeListener Class Reference

6.142.3.100 MamdaFieldState Wombat::MamdaTradeListener::getPartIdFieldState () const [virtual]

Get the participant identifier field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.142.3.101 MamdaFieldState Wombat::MamdaTradeListener::getSrcTimeFieldState () const [virtual]

Get the source time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.142.3.102 MamdaFieldState Wombat::MamdaTradeListener::getActivityTimeFieldState () const [virtual]

Get the activity time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.142.3.103 MamdaFieldState Wombat::MamdaTradeListener::getLineTimeFieldState () const [virtual]

Get the line time of the update.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.
6.142.3.104 MamdaFieldState Wombat::MamdaTradeListener::getSendTimeFieldState () const [virtual]

Get the send time field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.142.3.105 MamdaFieldState Wombat::MamdaTradeListener::getMsgQualifierFieldState () const [virtual]

Get the message qualifier field state.

Returns:
Message Qualifier. The message qualifier provides information in relation to messages duplicate, delayed or out-of-sequence status.

Implements Wombat::MamdaBasicEvent.

6.142.3.106 MamdaFieldState Wombat::MamdaTradeListener::getPubIdFieldState () const

6.142.3.107 MamdaFieldState Wombat::MamdaTradeListener::getEventSeqNumFieldState () const [virtual]

The event SeqNum Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.108 MamdaFieldState Wombat::MamdaTradeListener::getEventTimeFieldState () const [virtual]

Get the event time field state.
Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaBasicEvent.

6.142.3.109 MamdaFieldState Wombat::MamdaTradeListener::getLastPriceFieldState () const [virtual]
The last trade price Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.110 MamdaFieldState Wombat::MamdaTradeListener::getLastVolumeFieldState () const [virtual]
The last volume Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.111 MamdaFieldState Wombat::MamdaTradeListener::getLastPartIdFieldState () const [virtual]
The last part Id Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.112 MamdaFieldState Wombat::MamdaTradeListener::getLastTimeFieldState () const [virtual]
The last time Field State.
Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.113 MamdaFieldState Wombat::MamdaTradeListener::getIrregPriceFieldState () const [virtual]

The irreg price Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.114 MamdaFieldState Wombat::MamdaTradeListener::getIrregVolumeFieldState () const [virtual]

The irreg volume Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.115 MamdaFieldState Wombat::MamdaTradeListener::getIrregPartIdFieldState () const [virtual]

The irreg part Id Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.116 MamdaFieldState Wombat::MamdaTradeListener::getIrregTimeFieldState () const [virtual]

The irregular time Field State.
6.142 Wombat::MamdaTradeListener Class Reference

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.117 MamdaFieldState Wombat::MamdaTradeListener::getTradeDateFieldState () const [virtual]

The trade date Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.118 MamdaFieldState Wombat::MamdaTradeListener::getSideFieldState () const [virtual]

The TradeSide or AggressorSide Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.119 MamdaFieldState Wombat::MamdaTradeListener::getTradeCountFieldState () const

6.142.3.120 MamdaFieldState Wombat::MamdaTradeListener::getAccVolumeFieldState () const [virtual]

The accumulated volume Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.
6.142.3.121 MamdaFieldState Wombat::MamdaTradeListener::getOffExAccVolumeFieldState () const
[virtual]

The off exchange accumulated volume Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.122 MamdaFieldState Wombat::MamdaTradeListener::getOnExAccVolumeFieldState () const
[virtual]

The on exchange accumulated volume Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.123 MamdaFieldState Wombat::MamdaTradeListener::getNetChangeFieldState () const
[virtual]

The net change Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.124 MamdaFieldState Wombat::MamdaTradeListener::getPctChangeFieldState () const
[virtual]

The percentage change Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.
6.142.3.125 \textbf{MamdaFieldState} Wombat::MamdaTradeListener::getTradeDirectionFieldState () const [virtual]

The trade direction Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements \texttt{Wombat::MamdaTradeRecap}.

6.142.3.126 \textbf{MamdaFieldState} Wombat::MamdaTradeListener::getOpenPriceFieldState () const [virtual]

The open price Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements \texttt{Wombat::MamdaTradeRecap}.

6.142.3.127 \textbf{MamdaFieldState} Wombat::MamdaTradeListener::getHighPriceFieldState () const [virtual]

The high price Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements \texttt{Wombat::MamdaTradeRecap}.

6.142.3.128 \textbf{MamdaFieldState} Wombat::MamdaTradeListener::getLowPriceFieldState () const [virtual]

The low price Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements \texttt{Wombat::MamdaTradeRecap}.
6.142.3.129 **MamdaFieldState** Wombat::MamdaTradeListener::getClosePriceFieldState () const
[virtual]

Get the field state.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaTradeClosing**.

6.142.3.130 **MamdaFieldState** Wombat::MamdaTradeListener::getPrevClosePriceFieldState () const
[virtual]

The previous close price Field State.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaTradeRecap**.

6.142.3.131 **MamdaFieldState** Wombat::MamdaTradeListener::getAdjPrevClosePriceFieldState () const
[virtual]

The adjusted previous close date Field State.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaTradeRecap**.

6.142.3.132 **MamdaFieldState** Wombat::MamdaTradeListener::getPrevCloseDateFieldState () const
[virtual]

The previous close date Field State.

**Returns:**
MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaTradeRecap**.
6.142.3.133 MamdaFieldState Wombat::MamdaTradeListener::getBlockCountFieldState () const  
[virtual]

The block count Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.134 MamdaFieldState Wombat::MamdaTradeListener::getBlockVolumeFieldState () const  
[virtual]

The block volume Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.135 MamdaFieldState Wombat::MamdaTradeListener::getVwapFieldState () const  
[virtual]

The Vwap Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.136 MamdaFieldState Wombat::MamdaTradeListener::getOffExVwapFieldState () const  
[virtual]

The off exchange Vwap Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.
6.142.3.137 MamdaFieldState Wombat::MamdaTradeListener::getOnExVwapFieldState () const  
[virtual]

The on exchange Vwap Field State.

Returns:
　MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.138 MamdaFieldState Wombat::MamdaTradeListener::getTotalValueFieldState () const  
[virtual]

The total value Field State.

Returns:
　MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.139 MamdaFieldState Wombat::MamdaTradeListener::getOffExTotalValueFieldState () const  
[virtual]

The Off Exchange Total Value Field State.

Returns:
　MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.140 MamdaFieldState Wombat::MamdaTradeListener::getOnExTotalValueFieldState () const  
[virtual]

The On Exchange Total Value Field State.

Returns:
　MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.
6.142.3.141  MamdaFieldState Wombat::MamdaTradeListener::getStdDevFieldState () const [virtual]

The std deviation Field State.

Returns:
    MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.142  MamdaFieldState Wombat::MamdaTradeListener::getStdDevSumFieldState () const [virtual]

The std deviation sum Field State.

Returns:
    MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.143  MamdaFieldState Wombat::MamdaTradeListener::getStdDevSumSquaresFieldState () const [virtual]

The StdDevSumSquares Field State.

Returns:
    MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.144  MamdaFieldState Wombat::MamdaTradeListener::getTradeUnitsFieldState () const [virtual]

The trade units Field State.

Returns:
    MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.
6.142.3.145 MamdaFieldState Wombat::MamdaTradeListener::getLastSeqNumFieldState () const
[virtual]

The last SeqNum Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.146 MamdaFieldState Wombat::MamdaTradeListener::getHighSeqNumFieldState () const
[virtual]

The high seqNum Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.147 MamdaFieldState Wombat::MamdaTradeListener::getLowSeqNumFieldState () const
[virtual]

The low SeqNum Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.148 MamdaFieldState Wombat::MamdaTradeListener::getTotalVolumeSeqNumFieldState () const
[virtual]

The total volume seqNum Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.
**6.142.3.149 MamdaFieldState** Wombat::MamdaTradeListener::getCurrencyCodeFieldState () const

[virtual]

The currency code Field State.

**Returns:**

  MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

**6.142.3.150 MamdaFieldState** Wombat::MamdaTradeListener::getSettlePriceFieldState () const

[virtual]

The settle price Field State.

**Returns:**

  MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

**6.142.3.151 MamdaFieldState** Wombat::MamdaTradeListener::getSettleDateFieldState () const

[virtual]

The last trade price Field State.

**Returns:**

  MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

**6.142.3.152 MamdaFieldState** Wombat::MamdaTradeListener::getTradeExecVenuFieldState () const

**6.142.3.153 MamdaFieldState** Wombat::MamdaTradeListener::getOffExchangeTradePriceFieldState () const

[virtual]

The settle date Field State.
Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.154 MamdaFieldState Wombat::MamdaTradeListener::getOnExchangeTradePriceFieldState () const
[virtual]

The onExchange trade price Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.155 MamdaFieldState Wombat::MamdaTradeListener::getTradePriceFieldState () const
[virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeOutOfSequence.

6.142.3.156 MamdaFieldState Wombat::MamdaTradeListener::getTradeVolumeFieldState () const
[virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeOutOfSequence.
6.142.3.157  MamdaFieldState Wombat::MamdaTradeListener::getTradePartIdFieldState () const  
            [virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeOutOfSequence.

6.142.3.158  MamdaFieldState Wombat::MamdaTradeListener::getTradeQualFieldState () const  
            [virtual]

The trade qual Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.159  MamdaFieldState Wombat::MamdaTradeListener::getTradeQualNativeFieldState () const  
            [virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeOutOfSequence.

6.142.3.160  MamdaFieldState Wombat::MamdaTradeListener::getTradeSellersSaleDaysFieldState () const  
            [virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeOutOfSequence.
6.142.3.161 MamdaFieldState Wombat::MamdaTradeListener::getTradeStopStockFieldState () const [virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeOutOfSequence.

6.142.3.162 MamdaFieldState Wombat::MamdaTradeListener::getIsIrregularFieldState () const [virtual]

The isIrregular Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.163 MamdaFieldState Wombat::MamdaTradeListener::getOrderIdFieldState () const [virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeOutOfSequence.

6.142.3.164 MamdaFieldState Wombat::MamdaTradeListener::getUniqueIdFieldState () const [virtual]

The unique ID Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeReport.
6.142.3.165 **MamdaFieldState** Wombat::MamdaTradeListener::getTradeIdFieldState () const [virtual]

The trade ID Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaTradeReport**.

6.142.3.166 **MamdaFieldState** Wombat::MamdaTradeListener::getCorrTradeIdFieldState () const [virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaTradeCorrection**.

6.142.3.167 **MamdaFieldState** Wombat::MamdaTradeListener::getTradeActionFieldState () const [virtual]

The trade action Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements **Wombat::MamdaTradeReport**.

6.142.3.168 **MamdaFieldState** Wombat::MamdaTradeListener::getTradeExecVenueFieldState () const

6.142.3.169 **MamdaFieldState** Wombat::MamdaTradeListener::getBeginGapSeqNumFieldState () const [virtual]

**Returns:**

The Field State of the start of the sequence number gap.
Implements Wombat::MamdaTradeGap.

### MamdaFieldState Wombat::MamdaTradeListener::getEndGapSeqNumFieldState () const

[virtual]

**Returns:**

The field state of the end of the sequence number gap.

Implements Wombat::MamdaTradeGap.

### MamdaFieldState Wombat::MamdaTradeListener::getIsCancelFieldState () const

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCancelOrError.

### MamdaFieldState Wombat::MamdaTradeListener::getOrigPriceFieldState () const

[virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCancelOrError.

### MamdaFieldState Wombat::MamdaTradeListener::getOrigVolumeFieldState () const

[virtual]

Get the field state.
Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.175 MamdaFieldState Wombat::MamdaTradeListener::getOrigPartIdFieldState () const [virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.176 MamdaFieldState Wombat::MamdaTradeListener::getOrigQualFieldState () const [virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.177 MamdaFieldState Wombat::MamdaTradeListener::getOrigQualNativeFieldState () const [virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.178 MamdaFieldState Wombat::MamdaTradeListener::getOrigSellersSaleDaysFieldState () const [virtual]

Get the field state.
Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.179 MamdaFieldState Wombat::MamdaTradeListener::getOrigStopStockFieldState () const [virtual]
Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.180 MamdaFieldState Wombat::MamdaTradeListener::getCorrPriceFieldState () const [virtual]
Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCorrection.

6.142.3.181 MamdaFieldState Wombat::MamdaTradeListener::getCorrVolumeFieldState () const [virtual]
Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCorrection.

6.142.3.182 MamdaFieldState Wombat::MamdaTradeListener::getCorrPartIdFieldState () const [virtual]
Get the field state.
Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCorrection.

6.142.3.183  

MamdaFieldState Wombat::MamdaTradeListener::getCorrQualFieldState () const  

[virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCorrection.

6.142.3.184  

MamdaFieldState Wombat::MamdaTradeListener::getCorrQualNativeFieldState () const  

[virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCorrection.

6.142.3.185  

MamdaFieldState Wombat::MamdaTradeListener::getCorrSellersSaleDaysFieldState () const  

[virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCorrection.

6.142.3.186  

MamdaFieldState Wombat::MamdaTradeListener::getCorrStopStockFieldState () const  

[virtual]

Get the field state.
Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeCorrection.

6.142.3.187 MamdaFieldState Wombat::MamdaTradeListener::getIsIndicativeFieldState () const

The trade count Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.

6.142.3.189 MamdaFieldState Wombat::MamdaTradeListener::getOrigTradeIdFieldState () const
[virtual]

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.190 MamdaFieldState Wombat::MamdaTradeListener::getGenericFlagFieldState () const
[virtual]

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.191 MamdaFieldState Wombat::MamdaTradeListener::getShortSaleCircuitBreakerFieldState () const
[virtual]

Returns:
The ShortSaleCircuitBreaker Field State. An enumeration representing field state.

Implements Wombat::MamdaTradeRecap.
6.142 Wombat::MamdaTradeListener Class Reference

6.142.3.192 MamdaFieldState Wombat::MamdaTradeListener::get-OrigShortSaleCircuitBreakerFieldState () const [virtual]

Returns:

The OrigShortSaleCircuitBreaker Field State.

Implements Wombat::MamdaTradeCancelOrError.

6.142.3.193 MamdaFieldState Wombat::MamdaTradeListener::get-CorrShortSaleCircuitBreakerFieldState () const [virtual]

Returns:

The OrigShortSaleCircuitBreaker Field State.

Implements Wombat::MamdaTradeCorrection.

6.142.3.194 virtual void Wombat::MamdaTradeListener::onMsg (MamdaSubscription * subscription, const MamaMsg & msg, short msgType) [virtual]

Implementation of MamdaListener interface.

Implements Wombat::MamdaMsgListener.

6.142.3.195 void Wombat::MamdaTradeListener::assertEqual (MamdaTradeListener * listener)

6.142.3.196 void Wombat::MamdaTradeListener::reset (void)

The documentation for this class was generated from the following file:

• MamdaTradeListener.h
6.143 Wombat::MamdaTradeOutOfSequence Class Reference

MamdaTradeOutOfSequence is an interface that provides access to fields related to trade updates which have been identified as being out of sequence with previous update (e.g.

```cpp
#include <MamdaTradeOutOfSequence.h>
```

Inheritance diagram for Wombat::MamdaTradeOutOfSequence::

```
Wombat::MamdaBasicEvent
      ▼
    Wombat::MamdaTradeOutOfSequence
          ▼
        Wombat::MamdaTradeListener
```

Public Member Functions

- virtual const MamaPrice & `getTradePrice ()` const =0
  
  Get the trade price.

- virtual MamdaFieldState `getTradePriceFieldState ()` const =0
  
  Get the field state.

- virtual mama_quantity_t `getTradeVolume ()` const =0
  
  Get the trade volume.

- virtual MamdaFieldState `getTradeVolumeFieldState ()` const =0
  
  Get the field state.

- virtual const char * `getTradePartId ()` const =0
  
  Get the participant identifier for the trade.

- virtual MamdaFieldState `getTradePartIdFieldState ()` const =0
  
  Get the field state.

- virtual const char * `getTradeQual ()` const =0
  
  Get the trade qualifier.
• virtual MamdaFieldState getTradeQualFieldState () const =0
  Get the field state.

• virtual const char * getTradeQualNative () const =0
  Get the Trade condition ("sale condition").

• virtual MamdaFieldState getTradeQualNativeFieldState () const =0
  Get the field state.

• virtual mamau32_t getTradeSellersSaleDays () const =0
  Get the seller’s sale days.

• virtual MamdaFieldState getTradeSellersSaleDaysFieldState () const =0
  Get the field state.

• virtual char getTradeStopStock () const =0
  Get the Stopped stock indicator.

• virtual MamdaFieldState getTradeStopStockFieldState () const =0
  Get the field state.

• virtual bool getIsIrregular () const =0
  Get whether or not the trade qualifies as an irregular trade.

• virtual MamdaFieldState getIsIrregularFieldState () const =0
  Get the field state.

• virtual mamau64_t getOrderId () const =0
  Get the trade’s order id, if available.

• virtual MamdaFieldState getOrderIdFieldState () const =0
  Get the field state.

• virtual ~MamdaTradeOutOfSequence ()

### 6.143.1 Detailed Description

`MamdaTradeOutOfSequence` is an interface that provides access to fields related to trade updates which have been identified as being out of sequence with previous update (e.g. retransmissions from exchange or a fault tolerant take over event on the feedhandlers.).
6.143.2 Constructor & Destructor Documentation

6.143.2.1 virtual Wombat::MamdaTradeOutOfSequence::~MamdaTradeOutOfSequence () [virtual]

374 {

6.143.3 Member Function Documentation

6.143.3.1 virtual const MamaPrice& Wombat::MamdaTradeOutOfSequence::getTradePrice () const [pure virtual]

Get the trade price.

Returns:

The monetary value of an individual share of the security at the time of the trade.

Implemented in Wombat::MamdaTradeListener.

6.143.3.2 virtual MamaFieldState Wombat::MamdaTradeOutOfSequence::getTradePriceFieldState () const [pure virtual]

Get the field state.

Returns:

MamaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.143.3.3 virtual mama_quantity_t Wombat::MamdaTradeOutOfSequence::getTradeVolume () const [pure virtual]

Get the trade volume.

Returns:

The number of shares traded in a single transaction for an individual security.

Implemented in Wombat::MamdaTradeListener.
6.143.3.4 virtual MamdaFieldState Wombat::MamdaTradeOutOfSequence::getTradeVolumeFieldState () const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.143.3.5 virtual const char∗ Wombat::MamdaTradeOutOfSequence::getTradePartId () const [pure virtual]

Get the participant identifier for the trade.
This is typically an exchange ID or a market maker ID. I

**Returns:**

Trade participant ID.

Implemented in Wombat::MamdaTradeListener.

6.143.3.6 virtual MamdaFieldState Wombat::MamdaTradeOutOfSequence::getTradePartIdFieldState () const [pure virtual]

Get the field state.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.143.3.7 virtual const char∗ Wombat::MamdaTradeOutOfSequence::getTradeQual () const [pure virtual]

Get the trade qualifier.
A normalized set of qualifiers for the current trade for the security. This field may contain multiple string values, separated by the colon character.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Regular trade. A trade made without stated conditions is deemed regular way for settlement on the third business day following the transaction date.</td>
</tr>
<tr>
<td>Acquisition</td>
<td>A transaction made on the Exchange as a result of an Exchange acquisition.</td>
</tr>
<tr>
<td>Bunched</td>
<td>A trade representing an aggregate of two or more regular trades in a security occurring at the same price either simultaneously or within the same 60 second period, with no individual trade exceeding 10,000 shares.</td>
</tr>
<tr>
<td>Cash</td>
<td>A transaction which calls for the delivery of securities and payment on the same day the trade takes place.</td>
</tr>
<tr>
<td>Distribution</td>
<td>Sale of a large block of stock in such a manner that the price is not adversely affected.</td>
</tr>
<tr>
<td>BunchedSold</td>
<td>A bunched trade which is reported late</td>
</tr>
<tr>
<td>Rule155</td>
<td>To qualify as a 155 print, a specialist arranges for the sale of the block at one &quot;clean-up&quot; price or at the different price limits on his book. If the block is sold at a &quot;clean-up&quot; price, the specialist should execute at the same price all the executable buy orders on his book. The sale qualifier is only applicable for AMEX trades.</td>
</tr>
<tr>
<td>SoldLast</td>
<td>Sold Last is used when a trade prints in sequence but is reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>NextDay</td>
<td>A transaction which calls for delivery of securities on the first business day after the trade date.</td>
</tr>
<tr>
<td>Opened</td>
<td>Indicates an opening transaction that is printed out of sequence or reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>PriorRef</td>
<td>An executed trade that relates to an obligation to trade at an earlier point in the trading day or that refer to a prior reference price. This may be the result of an order that was lost or misplaced or a SelectNet order that was not executed on a timely basis.</td>
</tr>
<tr>
<td>Seller</td>
<td>A Seller's Option transaction is a special transaction which gives the seller the right to deliver the stock at any time within a specific period, ranging from not less than four calendar days to no more than 60 calendar days.</td>
</tr>
<tr>
<td>SplitTrade</td>
<td>An execution in two markets when the specialist or Market Maker in the market first receiving the order agrees to execute that portion of the trade in his market, with no guarantee of price improvement.</td>
</tr>
</tbody>
</table>
Returns:

Trade qualifier.

Implemented in Wombat::MamdaTradeListener.

6.143.3.8 virtual MamdaFieldState Wombat::MamdaTradeOutOfSequence::getTradeQualFieldState () const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.143.3.9 virtual const char ∗ Wombat::MamdaTradeOutOfSequence::getTradeQualNative () const [pure virtual]

Get the Trade condition ("sale condition").

Feed-specific trade qualifier code(s). This field is provided primarily for completeness and/or troubleshooting.

Returns:

The Trade Condition.

See also:

getTradeQual()

Implemented in Wombat::MamdaTradeListener.

6.143.3.10 virtual MamdaFieldState Wombat::MamdaTradeOutOfSequence::getTradeQualNativeFieldState () const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.143.3.11 virtual mama_u32_t Wombat::MamdaTradeOutOfSequence::getTradeSellersSaleDays () const [pure virtual]

Get the seller's sale days.
Used when the trade qualifier is "Seller". Specifies the number of days that may elapse before delivery of the security.

Returns:
The Seller’s sale days.
Implemented in Wombat::MamdaTradeListener.

6.143.3.12 virtual MamdaFieldState Wombat::MamdaTradeOutOfSequence::getTradeSellersSaleDaysFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.
Implemented in Wombat::MamdaTradeListener.

6.143.3.13 virtual char Wombat::MamdaTradeOutOfSequence::getTradeStopStock () const [pure virtual]

Get the Stopped stock indicator.
Condition related to certain NYSE trading rules. This is not related to a halted security status. (0 == N/A; 1 == Applicable)

Returns:
Stopped stock indicator.
Implemented in Wombat::MamdaTradeListener.

6.143.3.14 virtual MamdaFieldState Wombat::MamdaTradeOutOfSequence::getTradeStopStockFieldState () const [pure virtual]

Get the field state.
Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.143.3.15 virtual bool Wombat::MamdaTradeOutOfSequence::getIsIrregular() const [pure virtual]

Get whether or not the trade qualifies as an irregular trade.
In general, only "regular" trades qualify to update the official last price and high/low prices.

Returns:
Whether the trade is irregular or not.

Implemented in Wombat::MamdaTradeListener.

6.143.3.16 virtual MamdaFieldState Wombat::MamdaTradeOutOfSequence::getIsIrregularFieldState() const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.143.3.17 virtual mama_u64_t Wombat::MamdaTradeOutOfSequence::getOrderId() const [pure virtual]

Get the trade’s order id, if available.

Returns:
The trade message unique order id number (if available).

Implemented in Wombat::MamdaTradeListener.
virtual MamdaFieldState Wombat::MamdaTradeOutOfSequence::getOrderIdFieldState() const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

The documentation for this class was generated from the following file:

- MamdaTradeOutOfSequence.h
MamdaTradePossiblyDuplicate is an interface that provides access to fields related to trade updates which have been identified as being possible duplicates of previous updates.

```
#include <MamdaTradePossiblyDuplicate.h>
```

Inheritance diagram for Wombat::MamdaTradePossiblyDuplicate::

```
Wombat::MamdaBasicEvent

Wombat::MamdaTradePossiblyDuplicate

Wombat::MamdaTradeListener
```

**Public Member Functions**

- virtual const MamaPrice & `getTradePrice` () const =0
  
  *Get the trade price.*

- virtual MamdaFieldState `getTradePriceFieldState` () const =0
  
  *Get the field state.*

- virtual mama_quantity_t `getTradeVolume` () const =0
  
  *Get the trade volume for this transaction.*

- virtual MamdaFieldState `getTradeVolumeFieldState` () const =0
  
  *Get the field state.*

- virtual const char * `getTradePartId` () const =0
  
  *Get the participant identifier for this trade.*

- virtual MamdaFieldState `getTradePartIdFieldState` () const =0
  
  *Get the field state.*

- virtual const char * `getTradeQual` () const =0
  
  *Get the field state.*
• virtual const char * getTradeQualNative() const =0
  Feed-specific trade qualifier code(s).

• virtual MamdaFieldState getTradeQualNativeFieldState() const =0
  Get the field state.

• virtual mama_u32_t getTradeSellersSaleDays() const =0
  Used when the trade qualifier is "Seller".

• virtual MamdaFieldState getTradeSellersSaleDaysFieldState() const =0
  Get the field state.

• virtual char getTradeStopStock() const =0
  Condition related to certain NYSE trading rules.

• virtual MamdaFieldState getTradeStopStockFieldState() const =0
  Get the field state.

• virtual bool getIsIrregular() const =0
  In general, only "regular" trades qualify to update the official last price and high/low prices.

• virtual MamdaFieldState getIsIrregularFieldState() const =0
  Get the field state.

• virtual mama_u64_t getOrderId() const =0
  Get the order id for the trade, if available.

• virtual MamdaFieldState getOrderIdFieldState() const =0
  Get the field state.

• virtual ~MamdaTradePossiblyDuplicate() 

6.144.1 Detailed Description

MamdaTradePossiblyDuplicate is an interface that provides access to fields related to trade updates which have been identified as being possible duplicates of previous updates.
6.144 Wombat::MamdaTradePossiblyDuplicate Class Reference

6.144.2 Constructor & Destructor Documentation

6.144.2.1 virtual Wombat::MamdaTradePossiblyDuplicate::~MamdaTradePossiblyDuplicate () [virtual]

372 {};

6.144.3 Member Function Documentation

6.144.3.1 virtual const MamaPrice& Wombat::MamdaTradePossiblyDuplicate::getTradePrice () const [pure virtual]

Get the trade price.

Returns:

The monetary value of an individual share of the security at the time of the trade.

Implemented in Wombat::MamdaTradeListener.

6.144.3.2 virtual MamaFieldState Wombat::MamdaTradePossiblyDuplicate::getTradePriceFieldState () const [pure virtual]

Get the field state.

Returns:

MamaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.144.3.3 virtual mama_quantity_t Wombat::MamdaTradePossiblyDuplicate::getTradeVolume () const [pure virtual]

Get the trade volume for this transaction.

Returns:

The number of shares traded in a single transaction for an individual security.

Implemented in Wombat::MamdaTradeListener.
6.144.3.4 virtual MamdaFieldState Wombat::MamdaTradePossiblyDuplicate::getTradeVolumeFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.144.3.5 virtual const char∗ Wombat::MamdaTradePossiblyDuplicate::getTradePartId () const [pure virtual]

Get the participant identifier for this trade.

Returns:
Trade participant ID. This is typically an exchange ID or a market maker ID.

Implemented in Wombat::MamdaTradeListener.

6.144.3.6 virtual MamdaFieldState Wombat::MamdaTradePossiblyDuplicate::getTradePartIdFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.144.3.7 virtual const char∗ Wombat::MamdaTradePossiblyDuplicate::getTradeQual () const [pure virtual]

Returns:
Trade qualifier. A normalized set of qualifiers for the current trade for the security. This field may contain multiple string values, separated by the colon(:) character. In the future, we will make this a configurable enumeration.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Regular trade. A trade made without stated conditions is deemed regular way for settlement on the third business day following the transaction date.</td>
</tr>
<tr>
<td>Acquisition</td>
<td>A transaction made on the Exchange as a result of an Exchange acquisition.</td>
</tr>
<tr>
<td>Bunched</td>
<td>A trade representing an aggregate of two or more regular trades in a security occurring at the same price either simultaneously or within the same 60 second period, with no individual trade exceeding 10,000 shares.</td>
</tr>
<tr>
<td>Cash</td>
<td>A transaction which calls for the delivery of securities and payment on the same day the trade takes place.</td>
</tr>
<tr>
<td>Distribution</td>
<td>Sale of a large block of stock in such a manner that the price is not adversely affected.</td>
</tr>
<tr>
<td>BunchedSold</td>
<td>A bunched trade which is reported late.</td>
</tr>
<tr>
<td>Rule155</td>
<td>To qualify as a 155 print, a specialist arranges for the sale of the block at one &quot;clean-up&quot; price or at the different price limits on his book. If the block is sold at a &quot;clean-up&quot; price, the specialist should execute at the same price all the executable buy orders on his book. The sale qualifier is only applicable for AMEX trades.</td>
</tr>
<tr>
<td>SoldLast</td>
<td>Sold Last is used when a trade prints in sequence but is reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>NextDay</td>
<td>A transaction which calls for delivery of securities on the first business day after the trade date.</td>
</tr>
<tr>
<td>Opened</td>
<td>Indicates an opening transaction that is printed out of sequence or reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>PriorRef</td>
<td>An executed trade that relates to an obligation to trade at an earlier point in the trading day or that refer to a prior reference price. This may be the result of an order that was lost or misplaced or a SelectNet order that was not executed on a timely basis.</td>
</tr>
<tr>
<td>Seller</td>
<td>An option transaction is a special transaction which gives the seller the right to deliver the stock at any time within a specific period, ranging from not less than four calendar days to no more than 60 calendar days.</td>
</tr>
<tr>
<td>SplitTrade</td>
<td>An execution in two markets when the specialist or Market Maker in the market first receiving the order crosses.</td>
</tr>
</tbody>
</table>
6.144.3.8  virtual MamdaFieldState Wombat::MamdaTradePossiblyDuplicate::getTradeQualFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.144.3.9  virtual const char ∗ Wombat::MamdaTradePossiblyDuplicate::getTradeQualNative () const [pure virtual]

Feed-specific trade qualifier code(s).
This field is provided primarily for completeness and/or troubleshooting.

Returns:
Trade condition (a.k.a. "sale condition").

See also:
getTradeQual()

Implemented in Wombat::MamdaTradeListener.

6.144.3.10 virtual MamdaFieldState Wombat::MamdaTradePossiblyDuplicate::getTradeQualNativeFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.144.3.11 virtual mama_u32_t Wombat::MamdaTradePossiblyDuplicate::getTradeSellersSaleDays () const [pure virtual]

Used when the trade qualifier is "Seller".
Specifies the number of days that may elapse before delivery of the security.

Returns:
    Seller's sale days.

Implemented in Wombat::MamdaTradeListener.

6.144.3.12 virtual MamdaFieldState Wombat::MamdaTradePossiblyDuplicate::getTradeSellersSaleDaysFieldState () const [pure virtual]

Get the field state.

Returns:
    MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.144.3.13 virtual char Wombat::MamdaTradePossiblyDuplicate::getTradeStopStock () const [pure virtual]

Condition related to certain NYSE trading rules.
This is not related to a halted security status. (0 == N/A; 1 == Applicable)

Returns:
    Stopped stock indicator.

Implemented in Wombat::MamdaTradeListener.

6.144.3.14 virtual MamdaFieldState Wombat::MamdaTradePossiblyDuplicate::getTradeStopStockFieldState () const [pure virtual]

Get the field state.
Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.144.3.15 virtual bool Wombat::MamdaTradePossiblyDuplicate::getIsIrregular () const [pure virtual]

In general, only "regular" trades qualify to update the official last price and high/low prices.

Returns:
Whether or not the trade qualifies as an irregular trade.

Implemented in Wombat::MamdaTradeListener.

6.144.3.16 virtual MamdaFieldState Wombat::MamdaTradePossiblyDuplicate::getIsIrregularFieldState () const [pure virtual]

Get the field state.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.144.3.17 virtual mama_u64_t Wombat::MamdaTradePossiblyDuplicate::getOrderId () const [pure virtual]

Get the order id for the trade, if available.

Returns:
The trade message unique order id number (if available).

Implemented in Wombat::MamdaTradeListener.
virtual MamdaFieldState Wombat::MamdaTradePossiblyDuplicate::getOrderIdFieldState () const [pure virtual]

Get the field state.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

The documentation for this class was generated from the following file:

- MamdaTradePossiblyDuplicate.h
MamdaTradeRecap is an interface that provides access to trade related fields.

#include <MamdaTradeRecap.h>

Inheritance diagram for Wombat::MamdaTradeRecap:

```
Wombat::MamdaBasicRecap
\|-- Wombat::MamdaTradeRecap
\|  \|-- Wombat::MamdaTradeListener
```

Public Member Functions

- virtual const MamaPrice & getLastPrice () const =0  
  Monetary value of an individual share of the security at the time of the trade.

- virtual mama_quantity_t getLastVolume () const =0  
  Number of shares traded in a single transaction for an individual security.

- virtual const char * getLastPartId () const =0  
  Trade participant ID.

- virtual const MamaDateTime & getLastTime () const =0  
  Time corresponding to the last trade, as reported by the feed.

- virtual const MamaPrice & getIrregPrice () const =0  
  Monetary value of an individual share of the security at the time of the last irregular trade.

- virtual mama_quantity_t getIrregVolume () const =0  
  Number of shares traded in a single transaction for an individual security.

- virtual const char * getIrregPartId () const =0  
  Irregular trade participant ID.

- virtual const MamaDateTime & getIrregTime () const =0  
  Time corresponding to the last irregular trade, as reported by the feed.
• virtual const MamaDateTime & getTradeDate () const =0
  Time corresponding to the last trade, as reported by the feed.

• virtual mama_u32_t getTradeCount () const =0
  The number of trades today.

• virtual mama_quantity_t getAccVolume () const =0
  Total volume of shares traded in a security at the time it is disseminated.

• virtual mama_quantity_t getOffExAccVolume () const =0
  Total volume of off-exchange shares traded in a security at the time it is disseminated.

• virtual mama_quantity_t getOnExAccVolume () const =0
  Total volume of on-exchange shares traded in a security at the time it is disseminated.

• virtual const MamaPrice & getNetChange () const =0
  Change in price compared with the previous closing price (i.e.

• virtual double getPctChange () const =0
  Percentage change in price compared with the previous closing price (i.e.

• virtual MamdaTradeDirection getTradeDirection () const =0
  Trade tick direction.

• virtual const MamaPrice & getOpenPrice () const =0
  The price of the first qualifying trade in the security during the current trading day.

• virtual const MamaPrice & getHighPrice () const =0
  Highest price paid for security during the trading day.

• virtual const MamaPrice & getLowPrice () const =0
  Lowest price paid for security during the trading day.

• virtual const MamaPrice & getClosePrice () const =0
  Today's closing price.

• virtual const MamaPrice & getPrevClosePrice () const =0
  The last qualifying trade price on the previous trading day.

• virtual const MamaPrice & getAdjPrevClosePrice () const =0
  The previous close price adjusted by corporate actions, such as dividends and stock splits on the ex-date.
• virtual const MamaDateTime & getPrevCloseDate () const =0
  Date corresponding to wPrevClosePrice.

• virtual mama_u32_t getBlockCount () const =0
  The number of block trades (at least 10,000 shares) today.

• virtual mama_quantity_t getBlockVolume () const =0
  Total volume of block trades today.

• virtual double getVwap () const =0
  Volume-weighted average price of a security at the time it is disseminated.

• virtual double getOffExVwap () const =0
  Volume-weighted average price of an off-exchange security at the time it is disseminated.

• virtual double getOnExVwap () const =0
  Volume-weighted average price of an on-exchange security at the time it is disseminated.

• virtual double getTotalValue () const =0
  Total value of all shares traded in a security at the time it is disseminated.

• virtual double getOffExTotalValue () const =0
  Total value of all off-exchange shares traded in a security at the time it is disseminated.

• virtual double getOnExTotalValue () const =0
  Total value of all on-exchange shares traded in a security at the time it is disseminated.

• virtual double getStdDev () const =0
  Standard deviation of last trade price of a security at the time it is disseminated.

• virtual double getStdDevSum () const =0
  Sum of the standard deviations.

• virtual double getStdDevSumSquares () const =0
  Square of the sum of the standard deviations.

• virtual const char * getTradeUnits () const =0
  Reuters trade units.

• virtual mama_seqnum_t getLastSeqNum () const =0
Sequence number of the last trade.

• virtual mama_seqnum_t getHighSeqNum () const =0
  
  Sequence number of incoming message which gives high value.

• virtual mama_seqnum_t getLowSeqNum () const =0
  
  Sequence number of incoming message which gives low value.

• virtual mama_seqnum_t getEventSeqNum () const =0
  
  Sequence number of trade.

• virtual mama_seqnum_t getTotalVolumeSeqNum () const =0
• virtual const char ∗ getCurrencyCode () const =0
  
  Currency of the trade (eg US$).

• virtual const MamaPrice & getSettlePrice () const =0
  
  Settle price of trade.

• virtual const MamaDateTime & getSettleDate () const =0
  
  Settle date of trade.

• virtual MamdaTradeExecVenue getTradeExecVenue () const =0
  
  Trade execution venue.

• virtual const MamaPrice & getOffExchangeTradePrice () const =0
  
  Monetary value of an individual share of the security off exchange at the time of the trade.

• virtual const MamaPrice & getOnExchangeTradePrice () const =0
  
  Monetary value of an individual share of the security on exchange at the time of the trade.

• virtual const char ∗ getSide () const =0
• virtual char getShortSaleCircuitBreaker () const =0
  
  get the ShortSaleCircuitBreaker

• virtual MamdaFieldState getLastPriceFieldState () const =0
  
  The last trade price Field State.

• virtual MamdaFieldState getLastVolumeFieldState () const =0
  
  The last volume Field State.

• virtual MamdaFieldState getLastPartIdFieldState () const =0
• virtual MamdaFieldState getLastTimeFieldState () const =0
  The last time Field State.

• virtual MamdaFieldState getIrregPriceFieldState () const =0
  The irreg price Field State.

• virtual MamdaFieldState getIrregVolumeFieldState () const =0
  The irreg volume Field State.

• virtual MamdaFieldState getIrregPartIdFieldState () const =0
  The irreg part Id Field State.

• virtual MamdaFieldState getIrregTimeFieldState () const =0
  The irregular time Field State.

• virtual MamdaFieldState getTradeDateFieldState () const =0
  The trade date Field State.

• virtual MamdaFieldState getTradeCountFieldState () const =0
  The trade count Field State.

• virtual MamdaFieldState getAccVolumeFieldState () const =0
  The accumulated volume Field State.

• virtual MamdaFieldState getOffExAccVolumeFieldState () const =0
  The off exchange accumulated volume Field State.

• virtual MamdaFieldState getOnExAccVolumeFieldState () const =0
  The on exchange accumulated volume Field State.

• virtual MamdaFieldState getNetChangeFieldState () const =0
  The net change Field State.

• virtual MamdaFieldState getPctChangeFieldState () const =0
  The percentage change Field State.

• virtual MamdaFieldState getTradeDirectionFieldState () const =0
  The trade direction Field State.

• virtual MamdaFieldState getOpenPriceFieldState () const =0
The open price Field State.

• virtual MamdaFieldState getHighPriceFieldState() const =0
  The high price Field State.

• virtual MamdaFieldState getLowPriceFieldState() const =0
  The low price Field State.

• virtual MamdaFieldState getClosePriceFieldState() const =0
  The close price Field State.

• virtual MamdaFieldState getPrevClosePriceFieldState() const =0
  The previous close price Field State.

• virtual MamdaFieldState getAdjPrevClosePriceFieldState() const =0
  The adjusted previous close date Field State.

• virtual MamdaFieldState getPrevCloseDateFieldState() const =0
  The previous close date Field State.

• virtual MamdaFieldState getBlockCountFieldState() const =0
  The block count Field State.

• virtual MamdaFieldState getBlockVolumeFieldState() const =0
  The block volume Field State.

• virtual MamdaFieldState getVwapFieldState() const =0
  The Vwap Field State.

• virtual MamdaFieldState getOffExVwapFieldState() const =0
  The off exchange Vwap Field State.

• virtual MamdaFieldState getOnExVwapFieldState() const =0
  The on exchange Vwap Field State.

• virtual MamdaFieldState getTotalValueFieldState() const =0
  The total value Field State.

• virtual MamdaFieldState getOffExTotalValueFieldState() const =0
  The Off Exchange Total Value Field State.

• virtual MamdaFieldState getOnExTotalValueFieldState() const =0
The On Exchange Total Value Field State.

- virtual MamdaFieldState getStdDevFieldState () const =0
  The std deviation Field State.

- virtual MamdaFieldState getStdDevSumFieldState () const =0
  The std deviation sum Field State.

- virtual MamdaFieldState getStdDevSumSquaresFieldState () const =0
  The StdDevSumSquares Field State.

- virtual MamdaFieldState getTradeUnitsFieldState () const =0
  The trade units Field State.

- virtual MamdaFieldState getLastSeqNumFieldState () const =0
  The last SeqNum Field State.

- virtual MamdaFieldState getHighSeqNumFieldState () const =0
  The high seqNum Field State.

- virtual MamdaFieldState getLowSeqNumFieldState () const =0
  The low SeqNum Field State.

- virtual MamdaFieldState getEventSeqNumFieldState () const =0
  The event SeqNum Field State.

- virtual MamdaFieldState getTotalVolumeSeqNumFieldState () const =0
  The total volume seqNum Field State.

- virtual MamdaFieldState getCurrencyCodeFieldState () const =0
  The currency code Field State.

- virtual MamdaFieldState getSettlePriceFieldState () const =0
  The settle price Field State.

- virtual MamdaFieldState getSettleDateFieldState () const =0
  The last trade price Field State.

- virtual MamdaFieldState getOffExchangeTradePriceFieldState () const =0
  The settle date Field State.

- virtual MamdaFieldState getOnExchangeTradePriceFieldState () const =0
The onExchange trade price Field State.

- virtual MamdaFieldState getSideFieldState() const =0
  The TradeSide or AggressorSide Field State.

- virtual MamdaFieldState getShortSaleCircuitBreakerFieldState() const =0
- virtual virtual ~MamdaTradeRecap() 

### 6.145.1 Detailed Description

MamdaTradeRecap is an interface that provides access to trade related fields.

### 6.145.2 Constructor & Destructor Documentation

#### 6.145.2.1 virtual Wombat::MamdaTradeRecap::~MamdaTradeRecap() [virtual]

720 {};

### 6.145.3 Member Function Documentation

#### 6.145.3.1 virtual const MamaPrice& Wombat::MamdaTradeRecap::getLastPrice() const [pure virtual]

Monetary value of an individual share of the security at the time of the trade.

**Returns:**

The last trade price.

Implemented in Wombat::MamdaTradeListener.

#### 6.145.3.2 virtual mama_quantity_t Wombat::MamdaTradeRecap::getLastVolume() const [pure virtual]

Number of shares traded in a single transaction for an individual security.

**Returns:**

The last trade volume.

Implemented in Wombat::MamdaTradeListener.
6.145.3.3 virtual const char* Wombat::MamdaTradeRecap::getLastPartId ()
const [pure virtual]

Trade participant ID.
This is typically an exchange ID, sometimes a market maker ID.

Returns:
The last trade participant identifier.
Implemented in Wombat::MamdaTradeListener.

6.145.3.4 virtual const MamaDateTime& Wombat::MamdaTradeRecap::getLastTime () const
[pure virtual]

Time corresponding to the last trade, as reported by the feed.
The exact time of the trade may not be available, since rules governing trade reporting allow for trades to be reported within a specified time limit.

Returns:
The last trade time.
Implemented in Wombat::MamdaTradeListener.

6.145.3.5 virtual const MamaPrice& Wombat::MamdaTradeRecap::getIrregPrice () const [pure virtual]

Monetary value of an individual share of the security at the time of the last irregular trade.

Returns:
The last irregular trade price.
Implemented in Wombat::MamdaTradeListener.

6.145.3.6 virtual mama_quantity_t Wombat::MamdaTradeRecap::getIrregVolume () const [pure virtual]

Number of shares traded in a single transaction for an individual security.
Returns:
The last irregular trade volume.

Implemented in Wombat::MamdaTradeListener.

6.145.3.7 virtual const char* Wombat::MamdaTradeRecap::getIrregPartId () const
[pure virtual]

Irregular trade participant ID.
This is typically an exchange ID, sometimes a market maker ID.

Returns:
The last irregular trade participant identifier.

Implemented in Wombat::MamdaTradeListener.

6.145.3.8 virtual const MamaDateTime& Wombat::MamdaTradeRecap::getIrregTime () const
[pure virtual]

Time corresponding to the last irregular trade, as reported by the feed.
The exact time of the trade may not be available, since rules governing trade reporting
allow for a trades to be reported within a specified time limit.

Returns:
The last irregular trade time.

Implemented in Wombat::MamdaTradeListener.

6.145.3.9 virtual const MamaDateTime& Wombat::MamdaTradeRecap::getTradeDate () const
[pure virtual]

Time corresponding to the last trade, as reported by the feed.
The exact time of the trade may not be available, since rules governing trade reporting
allow for a trades to be reported within a specified time limit.

Returns:
The last trade time.

Implemented in Wombat::MamdaTradeListener.
6.145.3.10 virtual mama_u32_t Wombat::MamdaTradeRecap::getTradeCount () const [pure virtual]

The number of trades today.

**Returns:**

The number of trades so far today.

Implemented in Wombat::MamdaTradeListener.

6.145.3.11 virtual mama_quantity_t Wombat::MamdaTradeRecap::getAccVolume () const [pure virtual]

Total volume of shares traded in a security at the time it is disseminated.

**Returns:**

Accumulated trade volume.

Implemented in Wombat::MamdaTradeListener.

6.145.3.12 virtual mama_quantity_t Wombat::MamdaTradeRecap::getOffExAccVolume () const [pure virtual]

Total volume of off-exchange shares traded in a security at the time it is disseminated.

**Returns:**

Accumulated off-exchange trade volume.

Implemented in Wombat::MamdaTradeListener.

6.145.3.13 virtual mama_quantity_t Wombat::MamdaTradeRecap::getOnExAccVolume () const [pure virtual]

Total volume of on-exchange shares traded in a security at the time it is disseminated.

**Returns:**

Accumulated on-exchange trade volume.

Implemented in Wombat::MamdaTradeListener.
6.145.3.14 virtual const MamaPrice& Wombat::MamdaTradeRecap::getNetChange () const [pure virtual]

Change in price compared with the previous closing price (i.e. previous closing price - trade price).

Returns:
Price change compared to previous price.

Implemented in Wombat::MamdaTradeListener.

6.145.3.15 virtual double Wombat::MamdaTradeRecap::getPctChange () const [pure virtual]

Percentage change in price compared with the previous closing price (i.e. previous closing price - trade price).

Returns:
Percentage price change compared to previous price.

Implemented in Wombat::MamdaTradeListener.

6.145.3.16 virtual MamdaTradeDirection Wombat::MamdaTradeRecap::getTradeDirection () const [pure virtual]

Trade tick direction.
See MamdaTradeDirection.h for details.

Returns:
The tick direction.

Implemented in Wombat::MamdaTradeListener.

6.145.3.17 virtual const MamaPrice& Wombat::MamdaTradeRecap::getOpenPrice () const [pure virtual]

The price of the first qualifying trade in the security during the current trading day.
Returns:

The opening price.

Implemented in Wombat::MamdaTradeListener.

6.145.3.18 virtual const MamaPrice& Wombat::MamdaTradeRecap::getHighPrice () const [pure virtual]

Highest price paid for security during the trading day.

Returns:

The highest trade price for the day.

Implemented in Wombat::MamdaTradeListener.

6.145.3.19 virtual const MamaPrice& Wombat::MamdaTradeRecap::getLowPrice () const [pure virtual]

Lowest price paid for security during the trading day.

Returns:

The lowest trade price for the day.

Implemented in Wombat::MamdaTradeListener.

6.145.3.20 virtual const MamaPrice& Wombat::MamdaTradeRecap::getClosePrice () const [pure virtual]

Today’s closing price.
The closing price field is populated when official closing prices are sent by the feed after the session close.

Returns:

The closing price for the day.

Implemented in Wombat::MamdaTradeListener.
virtual const MamaPrice& Wombat::MamdaTradeRecap::getPrevClosePrice () const [pure virtual]

The last qualifying trade price on the previous trading day.
This field may be copied from the close price field during the morning "roll" of records in the feedhandler, or it may be obtained from a secondary source, or it may be explicitly sent by the feed prior to the opening of trading for the current day.

Returns:
The last qualifying trade price on the previous trading day.

Implemented in Wombat::MamdaTradeListener.

virtual const MamaPrice& Wombat::MamdaTradeRecap::getAdjPrevClosePrice () const [pure virtual]

The previous close price adjusted by corporate actions, such as dividends and stock splits on the ex-date.

Returns:
The adjusted previous closing price.

See also:
getPrevClosePrice()

Implemented in Wombat::MamdaTradeListener.

virtual const MamaDateTime& Wombat::MamdaTradeRecap::getPrevCloseDate () const [pure virtual]

Date corresponding to wPrevClosePrice.

Returns:
The closing price from the previous trading day.

See also:
getPrevClosePrice().

Implemented in Wombat::MamdaTradeListener.
virtual mama_u32_t Wombat::MamdaTradeRecap::getBlockCount () const  [pure virtual]

The number of block trades (at least 10,000 shares) today.

Returns:
Number of block trades.

Implemented in Wombat::MamdaTradeListener.

virtual mama_quantity_t Wombat::MamdaTradeRecap::getBlockVolume () const  [pure virtual]

Total volume of block trades today.

Returns:
Total volume of block trades.

Implemented in Wombat::MamdaTradeListener.

virtual double Wombat::MamdaTradeRecap::getVwap () const  [pure virtual]

Volume-weighted average price of a security at the time it is disseminated.
Equivalent to dividing total value by total volume.

Returns:
The VWAP (Volume-weighted average price)

Implemented in Wombat::MamdaTradeListener.

virtual double Wombat::MamdaTradeRecap::getOffExVwap () const  [pure virtual]

Volume-weighted average price of an off-exchange security at the time it is disseminated.
Equivalent to dividing the off-exchange total value by the off-exchange total volume.

Returns:
The off-exchange VWAP (Volume-weighted average price)
6.145 Wombat::MamdaTradeRecap Class Reference

Implemented in Wombat::MamdaTradeListener.

6.145.3.28 virtual double Wombat::MamdaTradeRecap::getOnExVwap () const [pure virtual]

Volume-weighted average price of an on-exchange security at the time it is disseminated.
Equivalent to dividing on-exchange total value by the on-exchange total volume.

Returns:
The on-exchange VWAP (Volume-weighted average price)

Implemented in Wombat::MamdaTradeListener.

6.145.3.29 virtual double Wombat::MamdaTradeRecap::getTotalValue () const [pure virtual]

Total value of all shares traded in a security at the time it is disseminated.
Calculated by summing the result of multiplying the trade price by trade volume for each qualifying trade.

Returns:
Total value of all instruments traded.

Implemented in Wombat::MamdaTradeListener.

6.145.3.30 virtual double Wombat::MamdaTradeRecap::getOffExTotalValue () const [pure virtual]

Total value of all off-exchange shares traded in a security at the time it is disseminated.
Calculated by summing the result of multiplying the trade price by trade volume for each qualifying trade.

Returns:
Total value of all off-exchange instruments traded.

Implemented in Wombat::MamdaTradeListener.
6.145.3.31 virtual double Wombat::MamdaTradeRecap::getOnExTotalValue () const [pure virtual]

Total value of all on-exchange shares traded in a security at the time it is disseminated. Calculated by summing the result of multiplying the trade price by trade volume for each qualifying trade.

Returns:
Total value of all on-exchange instruments traded.

Implemented in Wombat::MamdaTradeListener.

6.145.3.32 virtual double Wombat::MamdaTradeRecap::getStdDev () const [pure virtual]

Standard deviation of last trade price of a security at the time it is disseminated.

Returns:
The standard deviation of last trade price.

Implemented in Wombat::MamdaTradeListener.

6.145.3.33 virtual double Wombat::MamdaTradeRecap::getStdDevSum () const [pure virtual]

Sum of the standard deviations.

Returns:
The sum of the standard deviations.

Implemented in Wombat::MamdaTradeListener.

6.145.3.34 virtual double Wombat::MamdaTradeRecap::getStdDevSumSquares () const [pure virtual]

Square of the sum of the standard deviations.

Returns:
The square of the sum of the standard deviations.

Implemented in Wombat::MamdaTradeListener.
6.145 Wombat::MamdaTradeRecap Class Reference

6.145.3.35 virtual const char* Wombat::MamdaTradeRecap::getTradeUnits () const [pure virtual]

Returns:
Reuters trade units.

Implemented in Wombat::MamdaTradeListener.

6.145.3.36 virtual mama_seqnum_t Wombat::MamdaTradeRecap::getLastSeqNum () const [pure virtual]

Returns:
The sequence number of the last trade.

Implemented in Wombat::MamdaTradeListener.

6.145.3.37 virtual mama_seqnum_t Wombat::MamdaTradeRecap::getHighSeqNum () const [pure virtual]

Returns:
Sequence number of incoming message which gives high value.

Implemented in Wombat::MamdaTradeListener.

6.145.3.38 virtual mama_seqnum_t Wombat::MamdaTradeRecap::getLowSeqNum () const [pure virtual]

Returns:
Sequence number of incoming message which gives low value.

Implemented in Wombat::MamdaTradeListener.
6.145.3.39 virtual mama_seqnum_t Wombat::MamdaTradeRecap::getEventSeqNum () const [pure virtual]

Sequence number of trade.

Returns:
Sequence number of trade.

Implemented in Wombat::MamdaTradeListener.

6.145.3.40 virtual mama_seqnum_t Wombat::MamdaTradeRecap::getTotalVolumeSeqNum () const [pure virtual]

Implemented in Wombat::MamdaTradeListener.

6.145.3.41 virtual const char* Wombat::MamdaTradeRecap::getCurrencyCode () const [pure virtual]

Currency of the trade (eg US$).

Returns:
Currency of the trade.

Implemented in Wombat::MamdaTradeListener.

6.145.3.42 virtual const MamaPrice& Wombat::MamdaTradeRecap::getSettlePrice () const [pure virtual]

Settle price of trade.

Returns:
Settle price of trade.

Implemented in Wombat::MamdaTradeListener.
6.145.3.43 virtual const MamaDateTime& Wombat::MamdaTradeRecap::getSettleDate () const [pure virtual]

Settle date of trade.

Returns:

Settle date of trade.

Implemented in Wombat::MamdaTradeListener.

6.145.3.44 virtual MamdaTradeExecVenue Wombat::MamdaTradeRecap::getTradeExecVenue () const [pure virtual]

Trade execution venue.

See MamdaTradeExecVenue.h for details.

Returns:

The trade execution venue.

Implemented in Wombat::MamdaTradeListener.

6.145.3.45 virtual const MamaPrice& Wombat::MamdaTradeRecap::getOffExchangeTradePrice () const [pure virtual]

Monetary value of an individual share of the security off exchange at the time of the trade.

Returns:

The last off exchange trade price.

Implemented in Wombat::MamdaTradeListener.

6.145.3.46 virtual const MamaPrice& Wombat::MamdaTradeRecap::getOnExchangeTradePrice () const [pure virtual]

Monetary value of an individual share of the security on exchange at the time of the trade.
Returns:

The last on exchange trade price.

Implemented in `Wombat::MamdaTradeListener`.

6.145.3.47 virtual const char* Wombat::MamdaTradeRecap::getSide () const
[pure virtual]

Returns:

the Aggressor Side or TradeSide TradeSide

- 0 : No TradeSide is currently known/available.
- 1 or B : Buy
- 2 or S : Sell
  - 0 : No AggressorSide is currently known/available.
  - 1 or B : Buy
  - 2 or S : Sell

Implemented in `Wombat::MamdaTradeListener`.

6.145.3.48 virtual char Wombat::MamdaTradeRecap::getShortSaleCircuitBreaker () const [pure virtual]

get the ShortSaleCircuitBreaker

Returns:

ShortSaleCircuitBreaker

- return values:
  - Blank: Short Sale Restriction Not in Effect.
  - A: Short Sale Restriction Activated.
  - C: Short Sale Restriction Continued.
  - D: Sale Restriction Deactivated.
  - E: Sale Restriction in Effect.

Implemented in `Wombat::MamdaTradeListener`.
6.145.3.49 virtual MamdaFieldState Wombat::MamdaTradeRecap::getLastPriceFieldState () const [pure virtual]

The last trade price Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.50 virtual MamdaFieldState Wombat::MamdaTradeRecap::getLastVolumeFieldState () const [pure virtual]

The last volume Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.51 virtual MamdaFieldState Wombat::MamdaTradeRecap::getLastPartIdFieldState () const [pure virtual]

The last part Id Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.52 virtual MamdaFieldState Wombat::MamdaTradeRecap::getLastTimeFieldState () const [pure virtual]

The last time Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.145.3.53 virtual MamdaFieldState Wombat::MamdaTradeRecap::getIrregPriceFieldState () const [pure virtual]

The irreg price Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.54 virtual MamdaFieldState Wombat::MamdaTradeRecap::getIrregVolumeFieldState () const [pure virtual]

The irreg volume Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.55 virtual MamdaFieldState Wombat::MamdaTradeRecap::getIrregPartIdFieldState () const [pure virtual]

The irreg part Id Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.56 virtual MamdaFieldState Wombat::MamdaTradeRecap::getIrregTimeFieldState () const [pure virtual]

The irregular time Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.145.3.57 virtual MamdaFieldState Wombat::MamdaTradeRecap::getTradeDateFieldState () const [pure virtual]

The trade date Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.58 virtual MamdaFieldState Wombat::MamdaTradeRecap::getTradeCountFieldState () const [pure virtual]

The trade count Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.59 virtual MamdaFieldState Wombat::MamdaTradeRecap::getAccVolumeFieldState () const [pure virtual]

The accumulated volume Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.60 virtual MamdaFieldState Wombat::MamdaTradeRecap::getOffExAccVolumeFieldState () const [pure virtual]

The off exchange accumulated volume Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.145.3.61 virtual MamdaFieldState Wombat::MamdaTradeRecap::getOnExAccVolumeFieldState () const [pure virtual]

The on exchange accumulated volume Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.62 virtual MamdaFieldState Wombat::MamdaTradeRecap::getNetChangeFieldState () const [pure virtual]

The net change Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.63 virtual MamdaFieldState Wombat::MamdaTradeRecap::getPctChangeFieldState () const [pure virtual]

The percentage change Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.64 virtual MamdaFieldState Wombat::MamdaTradeRecap::getTradeDirectionFieldState () const [pure virtual]

The trade direction Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
virtual MamdaFieldState Wombat::MamdaTradeRecap::getOpenPriceFieldState () const [pure virtual]

The open price Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getHighPriceFieldState () const [pure virtual]

The high price Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getLowPriceFieldState () const [pure virtual]

The low price Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getClosePriceFieldState () const [pure virtual]

The close price Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.145.3.69 virtual MamdaFieldState Wombat::MamdaTradeRecap::getPrevClosePriceFieldState () const [pure virtual]

The previous close price Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.70 virtual MamdaFieldState Wombat::MamdaTradeRecap::getAdjPrevClosePriceFieldState () const [pure virtual]

The adjusted previous close date Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.71 virtual MamdaFieldState Wombat::MamdaTradeRecap::getPrevCloseDateFieldState () const [pure virtual]

The previous close date Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.72 virtual MamdaFieldState Wombat::MamdaTradeRecap::getBlockCountFieldState () const [pure virtual]

The block count Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
virtual MamdaFieldState Wombat::MamdaTradeRecap::getBlockVolumeFieldState () const [pure virtual]

The block volume Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getVwapFieldState () const [pure virtual]

The Vwap Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getOffExVwapFieldState () const [pure virtual]

The off exchange Vwap Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getOnExVwapFieldState () const [pure virtual]

The on exchange Vwap Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
virtual MamdaFieldState Wombat::MamdaTradeRecap::getTotalValueFieldState() const [pure virtual]

The total value Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getOffExTotalValueFieldState() const [pure virtual]

The Off Exchange Total Value Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getOnExTotalValueFieldState() const [pure virtual]

The On Exchange Total Value Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getStdDevFieldState() const [pure virtual]

The std deviation Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.145 virtual MamdaFieldState Wombat::MamdaTradeRecap::getStdDevSumFieldState () const [pure virtual]

The std deviation sum Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145 virtual MamdaFieldState Wombat::MamdaTradeRecap::getStdDevSumSquaresFieldState () const [pure virtual]

The StdDevSumSquares Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145 virtual MamdaFieldState Wombat::MamdaTradeRecap::getTradeUnitsFieldState () const [pure virtual]

The trade units Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145 virtual MamdaFieldState Wombat::MamdaTradeRecap::getLastSeqNumFieldState () const [pure virtual]

The last SeqNum Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.145.3.85 virtual MamdaFieldState Wombat::MamdaTradeRecap::getHighSeqNumFieldState () const [pure virtual]

The high seqNum Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.86 virtual MamdaFieldState Wombat::MamdaTradeRecap::getLowSeqNumFieldState () const [pure virtual]

The low SeqNum Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.87 virtual MamdaFieldState Wombat::MamdaTradeRecap::getEventSeqNumFieldState () const [pure virtual]

The event SeqNum Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.88 virtual MamdaFieldState Wombat::MamdaTradeRecap::getTotalVolumeSeqNumFieldState () const [pure virtual]

The total volume seqNum Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
virtual MamdaFieldState Wombat::MamdaTradeRecap::getCurrencyCodeFieldState () const [pure virtual]

The currency code Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getSettlePriceFieldState () const [pure virtual]

The settle price Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getSettleDateFieldState () const [pure virtual]

The last trade price Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeRecap::getOffExchangeTradePriceFieldState () const [pure virtual]

The settle date Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.145.3.93 virtual MamdaFieldState Wombat::MamdaTradeRecap::getOnExchangeTradePriceFieldState () const [pure virtual]

The onExchange trade price Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.94 virtual MamdaFieldState Wombat::MamdaTradeRecap::getSideFieldState () const [pure virtual]

The TradeSide or AggressorSide Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.145.3.95 virtual MamdaFieldState Wombat::MamdaTradeRecap::getShortSaleCircuitBreakerFieldState () const [pure virtual]

Returns:

The ShortSaleCircuitBreaker Field State. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

The documentation for this class was generated from the following file:

- MamdaTradeRecap.h
MamdaTradeReport is an interface that provides access to fields related to a trade report.

#include <MamdaTradeReport.h>

Inheritance diagram for Wombat::MamdaTradeReport::

```
Wombat::MamdaBasicEvent

Wombat::MamdaTradeReport

Wombat::MamdaTradeListener
```

Public Member Functions

- virtual const MamaPrice & getTradePrice () const =0
  
  Get the trade price.

- virtual mama_quantity_t getTradeVolume () const =0
  
  Get the volume of shares in the trade.

- virtual const char * getTradePartId () const =0
  
  Get the participant identifier for the trade.

- virtual const char * getTradeQual () const =0
  
  Get the NYSE Technologies normalized trade qualifier.

- virtual const char * getTradeQualNative () const =0
  
  Get the native feed trade qualifier.

- virtual const char * getSide () const =0
- virtual MamdaFieldState getSideFieldState () const =0
  
  The TradeSide or AggressorSide Field State.

- virtual mama_u32_t getTradeSellersSaleDays () const =0
  
  Get the trade sellers sale days.

- virtual char getTradeStopStock () const =0
  
  Get the trade stop stock indicator.
• virtual bool getIsIrregular() const =0
  Get whether the trade is irregular.

• virtual mama_u64_t getOrderId() const =0
  Get the order id, if available.

• virtual const char * getUniqueId() const =0
  Get the unique ID.

• virtual const char * getTradeAction() const =0
  Get the trade action.

• virtual const char * getTradeId() const =0
  Get the trade id.

• virtual char getShortSaleCircuitBreaker() const =0
  Get the ShortSaleCircuitBreaker

• virtual MamdaFieldState getTradePriceFieldState() const =0
  The trade price Field State.

• virtual MamdaFieldState getTradeVolumeFieldState() const =0
  The trade volume Field State.

• virtual MamdaFieldState getTradePartIdFieldState() const =0
  The trade part ID Field State.

• virtual MamdaFieldState getTradeQualFieldState() const =0
  The trade qual Field State.

• virtual MamdaFieldState getTradeQualNativeFieldState() const =0
  The trade quality native Field State.

• virtual MamdaFieldState getTradeSellersSaleDaysFieldState() const =0
  The trade sellers sale days Field State.

• virtual MamdaFieldState getTradeStopStockFieldState() const =0
  The trade stop stock Field State.

• virtual MamdaFieldState getIsIrregularFieldState() const =0
  The isIrregular Field State.
6.146.1 Detailed Description

MamdaTradeReport is an interface that provides access to fields related to a trade report.
This class is used for all trade reports, whether those trades qualify as regular or irregular trades. (A regular trade generally qualifies to update the official last price and intraday high/low prices.)

6.146.2 Constructor & Destructor Documentation

6.146.2.1 virtual Wombat::MamdaTradeReport::~MamdaTradeReport () [virtual]

469 {};

6.146.3 Member Function Documentation

6.146.3.1 virtual const MamaPrice& Wombat::MamdaTradeReport::getTradePrice () const [pure virtual]

Get the trade price.

Returns:

The monetary value of an individual share of the security at the time of the trade.
6.146.3.2 virtual mama_quantity_t Wombat::MamdaTradeReport::getTradeVolume () const [pure virtual]

Get the volume of shares in the trade.

**Returns:**

The number of shares traded in a single transaction for an individual security.

Implemented in Wombat::MamdaTradeListener.

6.146.3.3 virtual const char ∗ Wombat::MamdaTradeReport::getTradePartId () const [pure virtual]

Get the participant identifier for the trade.

**Returns:**

Trade participant ID. This is typically an exchange ID or a market maker ID.

Implemented in Wombat::MamdaTradeListener.

6.146.3.4 virtual const char ∗ Wombat::MamdaTradeReport::getTradeQual () const [pure virtual]

Get the NYSE Technologies normalized trade qualifier.

**Returns:**

Trade qualifier. A normalized set of qualifiers for the current trade for the security. This field may contain multiple string values, separated by the colon(;) character.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>Regular trade. A trade made without stated conditions is deemed regular way for settlement on the third * business day following the transaction * date.</td>
</tr>
<tr>
<td>Acquisition</td>
<td>A transaction made on the Exchange as a result of an Exchange acquisition.</td>
</tr>
<tr>
<td>Bunched</td>
<td>A trade representing an aggregate of two or more regular trades in a security occurring at the same price either simultaneously or within the same 60 second period, with no individual trade exceeding 10,000 shares.</td>
</tr>
<tr>
<td>Cash</td>
<td>A transaction which calls for the delivery of securities and payment on the same day the trade takes place.</td>
</tr>
<tr>
<td>Distribution</td>
<td>Sale of a large block of stock in such a manner that the price is not adversely affected.</td>
</tr>
<tr>
<td>BunchedSold</td>
<td>A bunched trade which is reported late</td>
</tr>
<tr>
<td>Rule155</td>
<td>To qualify as a 155 print, a specialist arranges for the sale of the block at one &quot;clean-up&quot; price or at the different price limits on his book. If the block is sold at a &quot;clean-up&quot; price, the specialist should execute at the same price all the executable buy orders on his book. The sale qualifier is only applicable for AMEX trades.</td>
</tr>
<tr>
<td>SoldLast</td>
<td>Sold Last is used when a trade prints in sequence but is reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>NextDay</td>
<td>A transaction which calls for delivery of securities on the first business day after the trade date.</td>
</tr>
<tr>
<td>Opened</td>
<td>Indicates an opening transaction that is printed out of sequence or reported late or printed in conformance to the One or Two Point Rule.</td>
</tr>
<tr>
<td>PriorRef</td>
<td>An executed trade that relates to an obligation to trade at an earlier point in the trading day or that refer to a prior reference price. This may be the result of an order that was lost or misplaced or a SelectNet order that was not executed on a timely basis.</td>
</tr>
<tr>
<td>SplitTrade</td>
<td>An execution in two markets when the specialist or Market Maker in the market first receiving the order agrees to execute a portion of it at whatever manner that the price is not adversely affected.</td>
</tr>
</tbody>
</table>
Implemented in Wombat::MamdaTradeListener.

6.146.3.5 virtual const char ∗ Wombat::MamdaTradeReport::getTradeQualNative () const [pure virtual]

Get the native feed trade qualifier.

Returns:

Native trade qualifier (a.k.a. "sale condition"). Feed-specific trade qualifier code(s). This field is provided primarily for completeness and/or troubleshooting.

See also:

getTradeQual.

Implemented in Wombat::MamdaTradeListener.

6.146.3.6 virtual const char ∗ Wombat::MamdaTradeReport::getSide () const [pure virtual]

Returns:

the Aggressor Side or TradeSide TradeSide

• 0 : No TradeSide is currently known/available.
• 1 or B : Buy
• 2 or S : Sell
  – 0 : No AggressorSide is currently known/available.
  – 1 or B : Buy
  – 2 or S : Sell

Implemented in Wombat::MamdaTradeListener.

6.146.3.7 virtual MamdaFieldState Wombat::MamdaTradeReport::getSideFieldState () const [pure virtual]

The TradeSide or AggressorSide Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.146.3.8  virtual mama_u32_t Wombat::MamdaTradeReport::getTradeSellersSaleDays () const  [pure virtual]

Get the trade sellers sale days.

Returns:
   Seller's sale days. Used when the trade qualifier is "Seller". Specifies the number of days that may elapse before delivery of the security.

Implemented in Wombat::MamdaTradeListener.

6.146.3.9  virtual char Wombat::MamdaTradeReport::getTradeStopStock () const  [pure virtual]

Get the trade stop stock indicator.

Returns:
   Stopped stock indicator. Condition related to certain NYSE trading rules. This is not related to a halted security status. (0 == N/A; 1 == Applicable)

Implemented in Wombat::MamdaTradeListener.

6.146.3.10 virtual bool Wombat::MamdaTradeReport::getIsIrregular () const  [pure virtual]

Get whether the trade is irregular.

Returns:
   Whether or not the trade qualifies as an irregular trade. In general, only "regular" trades qualify to update the official last price and high/low prices.

Implemented in Wombat::MamdaTradeListener.

6.146.3.11 virtual mama_u64_t Wombat::MamdaTradeReport::getOrderId () const  [pure virtual]

Get the order id, if available.

Returns:
   The trade message unique order id number (if available).

Implemented in Wombat::MamdaTradeListener.
6.146.3.12 virtual const char* Wombat::MamdaTradeReport::getUniqueId () const [pure virtual]

Get the unique ID.

Returns:

The unique ID

Implemented in Wombat::MamdaTradeListener.

6.146.3.13 virtual const char* Wombat::MamdaTradeReport::getTradeAction () const [pure virtual]

Get the trade action.

Returns:

The trade action

Implemented in Wombat::MamdaTradeListener.

6.146.3.14 virtual const char* Wombat::MamdaTradeReport::getTradeId () const [pure virtual]

Get the trade id.

Returns:

the trade id

Implemented in Wombat::MamdaTradeListener.

6.146.3.15 virtual char Wombat::MamdaTradeReport::getShortSaleCircuitBreaker () const [pure virtual]

get the ShortSaleCircuitBreaker

Returns:

ShortSaleCircuitBreaker

• return values:
  • Blank: Short Sale Restriction Not in Effect.
• A: Short Sale Restriction Activated.
• C: Short Sale Restriction Continued.
• D: Sale Restriction Deactivated.
• E: Sale Restriction in Effect.

Implemented in `Wombat::MamdaTradeListener`.

### 6.146.3.16 virtual MamdaFieldState Wombat::MamdaTradeReport::getTradePriceFieldState () const [pure virtual]

The trade price Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaTradeListener`.

### 6.146.3.17 virtual MamdaFieldState Wombat::MamdaTradeReport::getTradeVolumeFieldState () const [pure virtual]

The trade volume Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaTradeListener`.

### 6.146.3.18 virtual MamdaFieldState Wombat::MamdaTradeReport::getTradePartIdFieldState () const [pure virtual]

The trade part ID Field State.

**Returns:**

MamdaFieldState. An enumeration representing field state.

Implemented in `Wombat::MamdaTradeListener`. 
6.146.3.19 virtual MamdaFieldState Wombat::MamdaTradeReport::getTradeQualFieldState () const [pure virtual]

The trade qual Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.146.3.20 virtual MamdaFieldState Wombat::MamdaTradeReport::getTradeQualNativeFieldState () const [pure virtual]

The trade quality native Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.146.3.21 virtual MamdaFieldState Wombat::MamdaTradeReport::getTradeSellersSaleDaysFieldState () const [pure virtual]

The trade sellers sale days Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.146.3.22 virtual MamdaFieldState Wombat::MamdaTradeReport::getTradeStopStockFieldState () const [pure virtual]

The trade stop stock Field State.

Returns:

MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
6.146.3.23 virtual MamdaFieldState Wombat::MamdaTradeReport::getIsIrregularFieldState () const

The isIrregular Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.146.3.24 virtual MamdaFieldState Wombat::MamdaTradeReport::getOrderIdFieldState () const

The order ID Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.146.3.25 virtual MamdaFieldState Wombat::MamdaTradeReport::getUniqueIdFieldState () const

The unique ID Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

6.146.3.26 virtual MamdaFieldState Wombat::MamdaTradeReport::getTradeActionFieldState () const

The trade action Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.
virtual MamdaFieldState Wombat::MamdaTradeReport::getTradeIdFieldState () const [pure virtual]

The trade ID Field State.

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

virtual MamdaFieldState Wombat::MamdaTradeReport::getShortSaleCircuitBreakerFieldState () const [pure virtual]

Returns:
MamdaFieldState. An enumeration representing field state.

Implemented in Wombat::MamdaTradeListener.

The documentation for this class was generated from the following file:

- MamdaTradeReport.h
6.147 map Class Reference

Inheritance diagram for map::

```
map
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Wombat::MamdaOptionContractSet</td>
</tr>
</tbody>
</table>
```

The documentation for this class was generated from the following files:

- MamdaOptionContractSet.h
- MamdaOptionExpirationDateSet.h
- MamdaOptionExpirationStrikes.h
Chapter 7

MAMDA C++ API File Documentation

7.1 MamdaAuctionFields.h File Reference

#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaAuctionFields
7.2 MamdaAuctionHandler.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaAuctionListener.h>
#include <mama/mamacpp.h>
#include <mamda/MamdaAuctionRecap.h>
#include <mamda/MamdaAuctionUpdate.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaAuctionHandler

  MamdaAuctionHandler is an interface for applications that want to have an easy way to access currency data.
MamdaAuctionListener is a class that specializes in handling currency data. Developers provide their own implementation of the MamdaAuctionHandler interface and will be delivered notifications for updates in the currency data.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicRecap.h>
#include <mamda/MamdaFieldState.h>
#include <mamda/MamdaUncrossPriceInd.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaAuctionRecap

  MamdaAuctionRecap is an interface that provides access to the currency related fields.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>
#include <mamda/MamdaUncrossPriceInd.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaAuctionUpdate

  MamdaAuctionUpdate is an interface that provides access to the currency related fields.
7.6 MamdaBasicEvent.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaFieldState.h>
#include <mama/mamacpp.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaBasicEvent

  MamdaBasicEvent is a superclass interface that provides access to common event related fields.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaFieldState.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaBasicRecap

  *MamdaBasicRecap* is an interface that provides access to recap related fields.
7.8 MamdaBasicSubscription.h File Reference

#include <mamda/MamdaConfig.h>
#include <mama/subscriptiontype.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaBasicSubscription
  
  A MamdaBasicSubscription is used to register interest in a particular symbol.
7.9 MamdaBookAtomicBookHandler.h File Reference

#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBookAtomicLevel.h>
#include <mamda/MamdaBookAtomicListener.h>
#include <mamda/MamdaBookAtomicGap.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaBookAtomicBookHandler
  
  MamdaBookAtomicBookHandler is an interface for applications that need to know when a MamdaBookAtomicListener finishes processing a single book update.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicEvent.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaBookAtomicGap
  
  **MamdaBookAtomicGap** is an interface that provides access to order book atomic update gap related fields.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicRecap.h>
#include <mamda/MamdaOrderBookTypes.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaBookAtomicLevel

  *MamdaBookAtomicLevel* is an interface that provides access to trade related fields.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicRecap.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaBookAtomicLevelEntry

  MamdaBookAtomicLevel is an interface that provides access to Price Level and Price Level Entry fields.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBookAtomicLevelEntry.h>
#include <mamda/MamdaBookAtomicListener.h>
#include <mamda/MamdaBookAtomicGap.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaBookAtomicLevelEntryHandler

  *MamdaBookAtomicLevelEntryHandler* is an interface for applications that want to have an easy way to handle order book Price Level & Entry updates.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBookAtomicLevel.h>
#include <mamda/MamdaBookAtomicListener.h>
#include <mamda/MamdaBookAtomicGap.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaBookAtomicLevelHandler

  *MamdaBookAtomicLevelHandler* is an interface for applications that want to have an easy way to handle order book Price Level updates.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaBookAtomicLevel.h>
#include <mamda/MamdaBookAtomicLevelEntry.h>
#include <mamda/MamdaBookAtomicGap.h>
#include <mamda/MamdaCommonFields.h>
#include <mamda/MamdaFieldState.h>
#include <mamda/MamdaOrderBookTypes.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaBookAtomicListener

  *MamdaBookAtomicListener* is a class that specializes in handling order book updates.
#include <mamda/MamdaCheckerType.h>

### Namespaces
- namespace Wombat

### Classes
- class Wombat::MamdaCheckerHandler
  
  `MamdaCheckerHandler` is an interface for applications that want to handle the results of the `MamdaQuoteChecker` and `MamdaTradeChecker`. 
# MamdaCheckerType.h File Reference

```c
#include <mama/log.h>
```

**Namespaces**

- namespace Wombat

**Enumerations**

- `enum Wombat::MamdaCheckerType { Wombat::MAMDA_CHECK_TYPE_NONE, Wombat::MAMDA_CHECK_TYPE_SNAPSHOT, Wombat::MAMDA_CHECK_TYPE_APPLY_DELTA }`

**Functions**

- `MAMAExpDLL const char * Wombat::mamdaCheckTypeToString (MamdaCheckerType type)`
7.18 MamdaCommonFields.h File Reference

#include <mamda/MamdaConfig.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaCommonFields

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing common fields from update messages.
7.19 MamdaConcreteBasicEvent.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaConcreteBasicEvent

MamdaConcreteBasicEvent is intended to be used to help implement concrete versions of various classes derived from MamdaBasicEvent.
# MamdaConfig.h File Reference

```cpp
#include <mama/config.h>
```
#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaCurrencyFields
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaCurrencyListener.h>
#include <mama/mamacpp.h>
#include <mamda/MamdaCurrencyRecap.h>
#include <mamda/MamdaCurrencyUpdate.h>

## Namespaces
- namespace Wombat

## Classes
- class Wombat::MamdaCurrencyHandler

*MamdaCurrencyHandler* is an interface for applications that want to have an easy way to access currency data.
7.23 MamdaCurrencyListener.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaSubscription.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaCurrencyRecap.h>
#include <mamda/MamdaCurrencyUpdate.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaCurrencyListener

  MamdaCurrencyListener is a class that specializes in handling currency data. Developers provide their own implementation of the MamdaCurrencyHandler interface and will be delivered notifications for updates in the currency data.
7.24 MamdaCurrencyRecap.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicRecap.h>
#include <mamda/MamdaFieldState.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaCurrencyRecap

  *MamdaCurrencyRecap is an interface that provides access to the currency related fields.*
7.25 MamdaCurrencyUpdate.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaCurrencyUpdate

  MamdaCurrencyUpdate is an interface that provides access to the currency related fields.
# MamdaDataException.h File Reference

```cpp
#include <mamda/MamdaConfig.h>
#include <stdexcept>
#include <string>
```

## Namespaces

- namespace Wombat

## Classes

- class Wombat::MamdaDataException

  MAMDA data exceptions.
#include <mamda/MamdaConfig.h>

**Namespaces**

- namespace Wombat

**Classes**

- class Wombat::MamdaErrorListener
  
  *MamdaErrorListener defines an interface for handling error notifications for a MamdaSubscription.*

- class Wombat::MamdaBasicErrorListener
  
  *MamdaBasicErrorListener defines an interface for handling error notifications for a MamdaBasicSubscription.*

**Enumerations**

- enum Wombat::MamdaErrorSeverity {
  
  Wombat::MAMDA_SEVERITY_OK,  
  Wombat::MAMDA_SEVERITY_LOW,  
  Wombat::MAMDA_SEVERITY_HIGH  
  }

  *MAMDA error severities are intended to provide a hint to the application as to the severity of an error.*

- enum Wombat::MamdaErrorCode {
  
  Wombat::MAMDA_ERROR_NO_ERROR,  
  Wombat::MAMDA_ERROR_BAD_SYMBOL,  
  Wombat::MAMDA_ERROR_EXPIRED,  
  Wombat::MAMDA_ERROR_TIME_OUT,  
  Wombat::MAMDA_ERROR_ENTITLEMENT,  
  Wombat::MAMDA_ERROR_NOT_FOUND,  
  Wombat::MAMDA_ERROR_DELETE  
  }

  *MAMDA error codes are currently a subset of the MAMA MsgStatus codes.*
#include <mamda/MamdaConfig.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaFields
#include <mamda/MamdaConfig.h>

**Namespaces**

- namespace Wombat

**Enumerations**

- enum Wombat::MamdaFieldState { Wombat::MODIFIED = 2, Wombat::NOT_MODIFIED = 1, Wombat::NOT_INITIALIZED = 0 }
  
  *An enumeration representing field state.*

**Functions**

- MAMDAExpDLL const char * Wombat::toString (MamdaFieldState fieldState)
  
  *Convert a MamdaFieldState to an appropriate, displayable string.*
7.30 MamdaFundamentalFields.h File Reference

#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaFundamentalFields
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaFundamentalListener.h>
#include <mama/mamacpp.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaFundamentalHandler

  MamdaFundamentalHandler is an interface for applications that want to have an
easy way to access fundamental equity pricing/analysis attributes, indicators and ra-
tios.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaSubscription.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaFundamentals.h>

**Namespaces**

- namespace Wombat

**Classes**

- class Wombat::MamdaFundamentalListener

  *MamdaFundamentalListener* is a class that specializes in handling fundamental equity pricing/analysis attributes, indicators and ratios.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicRecap.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaFundamentals
  
  *MamdaFundamentals* is an interface that provides access to the fundamental equity pricing/analysis attributes, indicators and ratios.
7.34 MamdaLock.h File Reference

#include <mamda/MamdaConfig.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaLock

Defines

• #define ACQUIRE_WLOCK(lock) (lock).acquire(MamdaLock::WRITE)
• #define ACQUIRE_RLOCK(lock) (lock).acquire(MamdaLock::READ)
• #define RELEASE_WLOCK(lock) (lock).release(MamdaLock::WRITE)
• #define RELEASE_RLOCK(lock) (lock).release(MamdaLock::READ)
• #define ACQUIRE_LOCK(lock, writeNeeded)
• #define RELEASE_LOCK(lock, isWrite)

7.34.1 Define Documentation

7.34.1.1 #define ACQUIRE_WLOCK(lock) (lock).acquire(MamdaLock::WRITE)

7.34.1.2 #define ACQUIRE_RLOCK(lock) (lock).acquire(MamdaLock::READ)

7.34.1.3 #define RELEASE_WLOCK(lock) (lock).release(MamdaLock::WRITE)

7.34.1.4 #define RELEASE_RLOCK(lock) (lock).release(MamdaLock::READ)

7.34.1.5 #define ACQUIRE_LOCK(lock, writeNeeded)

Value:

do {
    if (writeNeeded)
        (lock).acquire(MamdaLock::WRITE);
    else

(lock).acquire(MamdaLock::READ); \\ 
} while (0)

7.34.1.6  #define RELEASE_LOCK(lock, isWrite)

Value:

do { \\
  if (isWrite) \\
    (lock).release(MamdaLock::WRITE); \\
  else \\
    (lock).release(MamdaLock::READ); \\
} while (0)
Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaMsgListener
  
  MamdaMsgListener defines an interface for handling MAMA messages for a MamdaSubscription.

• class Wombat::MamdaBasicMsgListener
  
  MamdaBasicMsgListener defines an interface for handling MAMA messages for a MamdaBasicSubscription.
#include <mamda/MamdaConfig.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaMultiParticipantHandler

The MamdaMultiParticipantHandler class is an interface that allows a developer to be notified dynamically when participants are added to the list.
7.37 MamdaMultiParticipantManager.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaMultiParticipantHandler.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaMultiParticipantManager

MamdaMultiParticipantManager is a class that manages updates on a consolidated basis for securities that may be traded on multiple exchanges and which may have a national best bid and offer.
The MamdaMultiSecurityHandler class is an interface that allows a developer to be notified dynamically when securities are added to the list.
7.39 MamdaMultiSecurityManager.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaMultiSecurityHandler.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaMultiSecurityManager

MamdaMultiSecurityManager is a class that manages updates on an arbitrary number of securities that may be traded on multiple exchanges.
#include <mamda/MamdaOptionalConfig.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaNewsFields
# MamdaNewsHeadline.h File Reference

```cpp
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaNewsMetaData.h>
#include <mama/mamacpp.h>
```

## Namespaces

- namespace **Wombat**

## Classes

- class **Wombat::MamdaNewsHeadline**

  *MamdaNewsHeadline* represents a news headline and includes information about many types of meta-data attributes associated with the headline.
7.42 MamdaNewsHeadlineHandler.h File Reference

#include <mamda/MamdaOptionalConfig.h>
#include <mama/mamacpp.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaNewsHeadlineHandler

  *MamdaNewsHandler is an interface for applications that want to have an easy way to handle news headlines.*
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaErrorListener.h>
#include <mamda/MamdaQualityListener.h>
#include <mamda/MamdaNewsHeadline.h>
#include <mamda/MamdaNewsHeadlineHandler.h>
#include <mamda/MamdaNewsStory.h>
#include <mamda/MamdaStoryHandler.h>
#include <mamda/MamdaNewsQueryHandler.h>
#include <mamda/MamdaNewsQuery.h>

## Namespaces
- namespace Wombat

## Classes
- class Wombat::MamdaNewsManager

*MamdaNewsManager provides a class for managing access to streaming news headlines, headline queries, individual story queries, etc.*
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaNewsTypes.h>
#include <mama/types.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaNewsMetaData

MamdaNewsMetaData represents information about many types of attributes associated with the news headline.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaQuery.h>

Namespaces

- namespace Wombat

Enumerations

- enum Wombat::MamdaNewsQueryType
  
  Wombat::QUERY_TYPE_UNKNOWN = 0,
  Wombat::QUERY_TYPE_HISTORICAL = 1,
  Wombat::QUERY_TYPE_SUBSCRIPTION = 2,
  Wombat::QUERY_TYPE_HISTORICAL_SUBSCRIPTION = 3
#include <mamda/MamdaOptionalConfig.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaNewsQueryHandler

  MamdaNewsQueryHandler is an interface for applications that want to have an easy way to handle news queries.
## 7.47 MamdaNewsStory.h File Reference

```cpp
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaNewsMetaData.h>
#include <mama/mamacpp.h>
```

### Namespaces

- namespace Wombat

### Classes

- class Wombat::MamdaNewsStory

  *MamdaNewsStory* represents a complete text of a news story.
#include <mamda/MamdaOptionalConfig.h>
#include <mama/mamacpp.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaNewsStoryHandler

   MamdaNewsHandler is an interface for applications that want to have an easy way to handle news stories.
7.49 MamdaNewsTypes.h File Reference

#include <mamda/MamdaOptionalConfig.h>

Namespaces

• namespace Wombat

Typedefs

• typedef const char* Wombat::MamdaNewsStoryId
• typedef const char* Wombat::MamdaNewsHeadlineId

Enumerations

• enum Wombat::MamdaNewsPriority { Wombat::MAMDA_NEWS_PRIORITY_NONE = 0, Wombat::MAMDA_NEWS_PRIORITY_NORMAL = 5, Wombat::MAMDA_NEWS_PRIORITY_HOT = 9 }
#include <mama/config.h>
7.51  MamdaOptionAtTheMoneyCompareType.h  File Reference

Namespaces

• namespace Wombat

Enumerations

• enum Wombat::MamdaOptionAtTheMoneyCompareType
  { Wombat::MAMDA_AT_THE_MONEY_COMPARE_MID_QUOTE = 0,
    Wombat::MAMDA_AT_THE_MONEY_COMPARE_BID = 1,
    Wombat::MAMDA_AT_THE_MONEY_COMPARE_ASK = 2,
    Wombat::MAMDA_AT_THE_MONEY_COMPARE_LAST_TRADE = 3
  }

#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOptionAtTheMoneyCompareType.h>
#include <set>

## Namespaces

- namespace Wombat

## Classes

- class Wombat::MamdaOptionChain
  
  *MamdaOptionChain* is a specialized class to represent market data option chains.

- class Wombat::MamdaOptionChain::iterator
- class Wombat::MamdaOptionChain::const_iterator

## Typedefs

- typedef set< double > Wombat::StrikeSet
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOptionChainListener.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOptionChainHandler

Subclasses of this interface can be registered with the MamdaOptionChainListener in order to receive callbacks whenever the state of the underlying option chain changes on receipt of options updates.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaOptionSeriesUpdate.h>
#include <mamda/MamdaFieldState.h>

## Namespaces

- namespace Wombat

## Classes

- class Wombat::MamdaOptionChainListener
  
  MamdaOptionChainListener is a class that specializes in handling and managing option chain updates.
7.55 MamdaOptionChainView.h File Reference

#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOptionAtTheMoneyCompareType.h>
#include <mamda/MamdaOptionChainHandler.h>
#include <mamda/MamdaOptionExpirationDateSet.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaOptionChainView
  
  A class that represents a "view" of a subset of an option chain.
#include <mamda/MamdaOptionalConfig.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOptionChainViewRangeHandler

  Class to handle change in a MamdaOptionChainView range.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOptionTypes.h>
#include <mama/mamacpp.h>
#include <vector>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOptionContract
  
  A class that represents a single option contract.
#include <mamda/MamdaOptionalConfig.h>
#include <map>
#include <string.h>

Namespaces

- namespace Wombat

Classes

- struct Wombat::char_str_less_than
- class Wombat::MamdaOptionContractSet
  
  A class that represents a set of option contracts at a given strike price.
7.59  MamdaOptionExchangeUtils.h File Reference

#include <mamda/MamdaOptionalConfig.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOptionExchangeUtils
  A class with static utility functions for dealing with exchanges.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOptionAtTheMoneyCompareType.h>
#include <mamda/MamdaOptionExpirationStrikes.h>
#include <map>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOptionExpirationDateSet
  
  *A class that represents a set of expiration dates, each of which contains a set of strike prices, each of which contains a set of option contracts, each of which contains exchange-specific contracts.*
7.61 MamdaOptionExpirationStrikes.h File Reference

#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOptionAtTheMoneyCompareType.h>
#include <mamda/MamdaOptionStrikeSet.h>
#include <mamda/MamdaOptionChain.h>
#include <map>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaOptionExpirationStrikes

  A class that represents a set of strike prices at a particular expiration date.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaFields.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOptionFields

  Utility cache of MamaFieldDescriptors which are used internally by the API when accessing options related fields from update messages.
# MamdaOptionSeriesUpdate.h File Reference

```cpp
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicEvent.h>
```

## Namespaces

- namespace `Wombat`

## Classes

- class `Wombat::MamdaOptionSeriesUpdate`

  `MamdaOptionSeriesUpdate` is an interface that provides access to fields related to option series update events.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOptionContractSet.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOptionStrikeSet
  
  A class that represents the call and put contract sets at a given strike price.
7.65 MamdaOptionTypes.h File Reference

Namespaces

• namespace Wombat

Enumerations

• enum Wombat::MamdaOptionPutCall { Wombat::MAMDA_PUT_CALL_CALL = 'C', Wombat::MAMDA_PUT_CALL_PUT = 'P', Wombat::MAMDA_PUT_CALL_UNKNOWN = 'Z' }

  Enumeration for indicating whether an option contract is a put or a call.

• enum Wombat::MamdaOptionExerciseStyle { Wombat::MAMDA_EXERCISE_STYLE_AMERICAN = 'A', Wombat::MAMDA_EXERCISE_STYLE_EUROPEAN = 'E', Wombat::MAMDA_EXERCISE_STYLE_CAPPED = 'C', Wombat::MAMDA_EXERCISE_STYLE_UNKNOWN = 'Z' }

  Enumeration for indicating the style of an individual option contract.
#include <mamda/MamdaOrderBookEntry.h>
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOrderBookPriceLevel.h>
#include <mamda/MamdaOrderBookExceptions.h>
#include <mamda/MamdaOrderBookEntryFilter.h>
#include <mama/mamacpp.h>
#include <mama/MamaSourceDerivative.h>
#include <iosfwd>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBook
  
  *MamdaOrderBook* is a class that provides order book functionality, including iterators over price levels and entries within price levels.

- class Wombat::MamdaOrderBook::bidIterator
- class Wombat::MamdaOrderBook::askIterator
- class Wombat::MamdaOrderBook::bidEntryIterator
- class Wombat::MamdaOrderBook::askEntryIterator
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaOrderBookPriceLevel.h>
#include <mamda/MamdaOrderBookEntry.h>
#include <iosfwd>

**Namespaces**

- namespace Wombat

**Classes**

- class Wombat::MamdaOrderBookBasicDelta

  *MamdaOrderBookBasicDelta is a class that saves information about a basic order book delta.*
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOrderBookBasicDelta.h>
#include <iosfwd>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookBasicDeltaList
  
  MamdaOrderBookBasicDeltaList is a class that saves information about an order book delta that involves multiple entries and/or price levels.

- class Wombat::MamdaOrderBookBasicDeltaList::iterator
  
  The MamdaOrderBookBasicDeltaList's iterator provides access to the list of MamdaOrderBookBasicDelta objects that comprise it.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaSubscription.h>
#include <mamda/MamdaOrderBookCheckerHandler.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookChecker

`MamdaOrderBookChecker` is a class that provides order book sanity checking by periodically requesting snapshots of the order book from the publisher and comparing that with an order book being maintained in real time.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOrderBookCheckType.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookCheckerHandler

  *MamdaOrderBookCheckerHandler* is an interface for applications that want to handle the results of the *MamdaOrderBookChecker*. 
7.71  MamdaOrderBookCheckType.h File Reference

#include <mama/log.h>

Namespaces

• namespace Wombat

Enumerations

• enum Wombat::MamdaOrderBookCheckType { Wombat::MAMDA_BOOK_CHECK_TYPE_NONE = 0, Wombat::MAMDA_BOOK_CHECK_TYPE_SNAPSHOT = 1, Wombat::MAMDA_BOOK_CHECK_TYPE_APPLY_DELTA = 2 }

Functions

• MAMAExpDLL const char * Wombat::mamdaOrderBookCheckTypeToString (MamdaOrderBookCheckType type)
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaOrderBook.h>
#include <mamda/MamdaFieldState.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaOrderBookClear

   MamdaOrderBookClear is an interface that provides access to order book related fields.
Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookComplexDelta

  MamdaOrderBookComplexDelta is a class that saves information about a complex order book delta.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOrderBookComplexDelta.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookConcreteComplexDelta

  *MamdaOrderBookConcreteComplexDelta* is a class that saves information about a complex order book delta.
7.75 MamdaOrderBookConcreteSimpleDelta.h File Reference

#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOrderBookSimpleDelta.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookConcreteSimpleDelta
  
  *MamdaOrderBookConcreteSimpleDelta* is a class that saves information about a simple order book delta.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaOrderBook.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookDelta
  
  *MamdaOrderBookDelta* is an interface that provides access to order book related fields.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaOrderBook.h>
#include <mamda/MamdaOrderBookHandler.h>
#include <mamda/MamdaFieldState.h>

Classes

- class MamdaOrderBookDepthFilter
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOrderBookTypes.h>
#include <mamda/MamdaOrderBookPriceLevel.h>
#include <mamda/MamdaOrderBookExceptions.h>
#include <mama/MamaSource.h>
#include <mama/MamaSourceDerivative.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookEntry

  MamdaOrderBookEntry is a class that represents an entry within a price level of an order book.
7.79  MamdaOrderBookEntryFilter.h File Reference

#include <mamda/MamdaOptionalConfig.h>
#include <mama/MamaSource.h>
#include <iosfwd>

Namespaces

  • namespace Wombat

Classes

  • class Wombat::MamdaOrderBookEntryFilter
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOrderBookEntry.h>
#include <mamda/MamdaOrderBookExceptions.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaOrderBookEntryManager

MamdaOrderBookEntryManager is a class that provides a global order book lookup service, matching entry IDs that are unique across a set of order books.
#include <stdexcept>
#include <string>
#include <mamda/MamdaOptionalConfig.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookException
  
  *MamdaOrderBookException* is generated when an inconsistent state is detected in an order book.

- class Wombat::MamdaOrderBookDuplicateEntry
  
  *MamdaOrderBookDuplicateEntry* is generated when an existing entry is unexpectedly encountered when updating a *MamdaOrderBook* or *MamdaOrderBookEntryManager*.

- class Wombat::MamdaOrderBookMissingEntry
  
  *MamdaOrderBookMissingEntry* is generated when an expected entry is not found when updating a *MamdaOrderBook* or *MamdaOrderBookEntryManager*.

- class Wombat::MamdaOrderBookInvalidEntry
  
  *MamdaOrderBookInvalidEntry* is generated when an entry is applied as an update to or deletion from an order book but the internal references to price level and/or order book do not exist.
#include <mama/types.h>
#include <mamda/MamdaOptionalConfig.h>

## Namespaces

- namespace Wombat

## Classes

- class Wombat::MamdaOrderBookFields
  
  Utility cache of MamaFieldDescriptors which are used internally by the API when accessing orderbook related fields from update messages.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicEvent.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookGap

  _MamdaOrderBookGap_ is an interface that provides access to order book gap related fields.
MamdaOrderBookHandler is an interface for applications that want to have an easy way to handle order book updates.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaOrderBook.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookListener

  *MamdaOrderBookListener is a class that specializes in handling order book updates.*
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaOrderBookTypes.h>
#include <mamda/MamdaOrderBookEntryFilter.h>
#include <mama/mamacpp.h>
#include <stdlib.h>
#include <string.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookPriceLevel
  
  `MamdaOrderBookPriceLevel` is a class that provides a price level type for order books.

- class Wombat::MamdaOrderBookPriceLevel::iterator
7.87 MamdaOrderBookRecap.h File Reference

#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaOrderBook.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaOrderBookRecap

    MamdaOrderBookRecap is an interface that provides access to order book related fields.
#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaOrderBookBasicDelta.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaOrderBookSimpleDelta
  
  *MamdaOrderBookSimpleDelta* is a class that saves information about a simple order book delta.
#include <mamda/MamdaOptionalConfig.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderBookTypes

  *MamdaOrderBookTypes* is a class that provides order book related typed.
#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>

**Namespaces**

- namespace Wombat

**Classes**

- class Wombat::MamdaOrderImbalanceFields

  Utility cache of MamaFieldDescriptors which are used internally by the API when accessing imbalance related fields from update messages.
#include <mamda/MamdaConfig.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderImbalanceHandler

  `MamdaOrderImbalanceHandler` is an interface for applications that want to have an easy way to handle order imbalance updates.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaOrderImbalanceUpdate.h>
#include <mamda/MamdaOrderImbalanceRecap.h>
#include <mamda/MamdaFieldState.h>

## Namespaces

- namespace Wombat

## Classes

- class Wombat::MamdaOrderImbalanceListener
  
  A MamdaOrderImbalanceListener is class that specializes in handling an imbalance order updates.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicRecap.h>
#include <mamda/MamdaFieldState.h>
#include <mama/mamacpp.h>
#include <stdio.h>
#include <string.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderImbalanceRecap
#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaOrderImbalanceSide

Defines

• #define ASK_SIDE_IMBALANCE "ASK"
• #define BUY_SIDE_IMBALANCE "BID"
• #define NO_IMBALANCE_SIDE "NONE"

7.94.1 Define Documentation

7.94.1.1 #define ASK_SIDE_IMBALANCE "ASK"
7.94.1.2 #define BUY_SIDE_IMBALANCE "BID"
7.94.1.3 #define NO_IMBALANCE_SIDE "NONE"
#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>
#include <stdio.h>
#include <string.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaOrderImbalanceType

Defines

- #define MAMDA_MARKET_IMBALANCE_BUY "MktImbBuy"
- #define MAMDA_MARKET_IMBALANCE_SELL "MktImbSell"
- #define MAMDA_NO_MARKET_IMBALANCE "NoMktImb"
- #define MAMDA_MOC_IMBALANCE_BUY "MocImbBuy"
- #define MAMDA_MOC_IMBALANCE_SELL "MocImbSell"
- #define MAMDA_NO_MOC_IMBALANCE "NoMocImb"
- #define MAMDA_ORDER_IMB "OrderImb"
- #define MAMDA_ORDER_INF "OrderInf"
- #define MAMDA_ORDER_IMBALANCE_BUY "OrderImbBuy"
- #define MAMDA_ORDER_IMBALANCE_SELL "OrderImbSell"
- #define MAMDA_NO_ORDER_IMBALANCE "OrderImbNone"
- #define MAMDA_IMBALANCE_UNKNOWN "UNKNOWN"
7.95.1 Define Documentation

7.95.1.1 #define MAMDA_MARKET_IMBALANCE_BUY "MktImbBuy"

7.95.1.2 #define MAMDA_MARKET_IMBALANCE_SELL "MktImbSell"

7.95.1.3 #define MAMDA_NO_MARKET_IMBALANCE "NoMktImb"

7.95.1.4 #define MAMDA_MOC_IMBALANCE_BUY "MocImbBuy"

7.95.1.5 #define MAMDA_MOC_IMBALANCE_SELL "MocImbSell"

7.95.1.6 #define MAMDA_NO_MOC_IMBALANCE "NoMocImb"

7.95.1.7 #define MAMDA_ORDER_IMB "OrderImb"

7.95.1.8 #define MAMDA_ORDER_INF "OrderInf"

7.95.1.9 #define MAMDA_ORDER_IMBALANCE_BUY "OrderImbBuy"

7.95.1.10 #define MAMDA_ORDER_IMBALANCE_SELL "OrderImbSell"

7.95.1.11 #define MAMDA_NO_ORDER_IMBALANCE "OrderImbNone"

7.95.1.12 #define MAMDA_IMBALANCE_UNKNOWN "UNKNOWN"
7.96 MamdaOrderImbalanceUpdate.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaOrderImbalanceUpdate
# MamdaPubStatus.h File Reference

```cpp
#include <mamda/MamdaConfig.h>
```

## Namespaces

- namespace Wombat

## Classes

- class Wombat::MamdaPubStatus

  *MamdaPubStatus is an interface that provides access to the Security Status fields such as symbol announce messages.*
7.98 MamdaPubStatusFields.h File Reference

#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaPubStatusFields
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaPubStatusListener.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaPubStatusHandler

    MamdaPubStatusHandler is an interface for applications that want to have an easy way to handle feed handler publisher status updates.
7.100 MamdaPubStatusListener.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaPubStatus.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaPubStatusListener

MamdaPubStatusListener is a class that specializes in handling Publisher (Feed Handler) Status updates.
#include <mamda/MamdaConfig.h>
#include <mama/quality.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaQualityListener
  
  *MamdaQualityListener defines an interface for handling changes in quality notifications for a MamdaSubscription.*

- class Wombat::MamdaBasicQualityListener
  
  *MamdaBasicQualityListener defines an interface for handling changes in quality notifications for a MamdaBasicSubscription.*
7.102 MamdaQuery.h File Reference

#include <mamda/MamdaConfig.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaQuery
- class Wombat::MamdaOrQuery
- class Wombat::MamdaAndQuery
- class Wombat::MamdaEqualsQuery
- class Wombat::MamdaDateQuery
- class Wombat::MamdaContainsAllQuery
- class Wombat::MamdaContainsQuery
#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>
#include <mamda/MamdaCheckerHandler.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaQuoteChecker

  MamdaQuoteChecker is a class that provides quotes sanity checking by periodically requesting snapshots of the quotes from the publisher and comparing that with an quotes being maintained in real time.
7.104 MamdaQuoteClosing.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaQuoteClosing

   MamdaQuoteClosing is an interface that provides access to quote closing related fields.
#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaQuoteFields
  Utility cache of MamaFieldDescriptors which are used internally by the API when accessing quote related fields from update messages.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaQuoteGap

>MamdaQuoteGap is an interface that provides access to quote gap related fields.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaQuoteListener.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaQuoteHandler

  *MamdaQuoteHandler is an interface for applications that want to have an easy way to handle quote updates.*
7.108 MamdaQuoteListener.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaQuoteRecap.h>
#include <mamda/MamdaQuoteUpdate.h>
#include <mamda/MamdaQuoteGap.h>
#include <mamda/MamdaQuoteClosing.h>
#include <mamda/MamdaQuoteOutOfSequence.h>
#include <mamda/MamdaQuotePossiblyDuplicate.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaQuoteListener

   MamdaQuoteListener is a class that specializes in handling quote updates.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaQuoteOutOfSequence

  *MamdaQuoteOutOfSequence* is an interface that provides access to fields related to quote updates.
7.110 MamdaQuotePossiblyDuplicate.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaQuotePossiblyDuplicate

MamdaQuotePossiblyDuplicate is an interface that provides access to fields related to quote updates which are possible duplicates of previous quote updates.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicRecap.h>
#include <mamda/MamdaFieldState.h>
#include <mama/types.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaQuoteRecap

  *MamdaQuoteRecap* is an interface that provides access to quote related fields.
7.112 MamdaQuoteToBookListener.h File Reference

#include <mamda/MamdaOptionalConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaOrderBook.h>
#include <mamda/MamdaOrderBookHandler.h>
#include <mamda/MamdaFieldState.h>

Classes

• class MamdaQuoteToBookListener
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaQuoteUpdate

  *MamdaQuoteUpdate* is an interface that provides access to fields related to quote updates.
Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaSecStatus

  *MamdaSecStatus* is an interface that provides access to the Security Status fields such as symbol announce messages.
#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaSecStatusFields
  
  Utility cache of MamaFieldDescriptors which are used internally by the API when accessing security status related fields from update messages.
7.116 MamdaSecStatusHandler.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaSecStatusListener.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaSecStatusHandler

MamdaSecStatusHandler is an interface for applications that want to have an easy way to handle security status updates.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaSecStatus.h>
#include <mamda/MamdaSecStatusRecap.h>
#include <mamda/MamdaSecurityStatus.h>
#include <mamda/MamdaSecurityStatusQual.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaSecStatusListener

  MamdaSecurityStatusListener is a class that specializes in handling security status updates.
Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaSecStatusRecap

  MamdaSecStatus is an interface that provides access to the Security Status fields such as symbol announce messages.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaSymbolSourceEvent.h>
#include <mamda/MamdaSecStatusHandler.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaSecStatusSymbolSourceAdapter

MamdaSecStatusSymbolSourceAdapter is a simple adapter class that can be added as a handler to the MamdaSecStatusListener turning it into a MamdaSymbolSource.
7.120 MamdaSecurityStatus.h File Reference

#include <mamda/MamdaConfig.h>

Namespaces

• namespace Wombat

Enumerations

• enum Wombat::MamdaSecurityStatus {
    Wombat::SECURITY_STATUS_NONE,
    Wombat::SECURITY_STATUS_NORMAL,
    Wombat::SECURITY_STATUS_CLOSED,
    Wombat::SECURITY_STATUS_HALTED,
    Wombat::SECURITY_STATUS_NOT_EXIST,
    Wombat::SECURITY_STATUS_DELETED,
    Wombat::SECURITY_STATUS_AUCTION,
    Wombat::SECURITY_STATUS_CROSSING,
    Wombat::SECURITY_STATUS_SUSPENDED,
    Wombat::SECURITY_STATUS_AT_LAST,
    Wombat::SECURITY_STATUS_UNKNOWN = 99
} 

An enumeration representing the status of a security such as whether or not it is halted or closed for trading.

Functions

• MAMDAExpDLL const char * Wombat::toString (MamdaSecurityStatus securityStatus)

    Convert a MamdaSecurityStatus to an appropriate, displayable string.

• MAMDAExpDLL MamdaSecurityStatus Wombat::mamdaSecurityStatusFromString (const char *securityStatusStr)

    Convert a string representation of a security status to the enumeration.
#include <mamda/MamdaConfig.h>

Namespaces

- namespace Wombat

Enumerations

- enum Wombat::MamdaSecurityStatusQual

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SECURITY_STATUS_QUAL_NONE</td>
</tr>
<tr>
<td>1</td>
<td>SECURITY_STATUS_QUAL_OPENING</td>
</tr>
<tr>
<td>7</td>
<td>SECURITY_STATUS_QUAL_EXCUSED</td>
</tr>
<tr>
<td>8</td>
<td>SECURITY_STATUS_QUAL_WITHDRAWN</td>
</tr>
<tr>
<td>9</td>
<td>SECURITY_STATUS_QUAL_SUSPENDED</td>
</tr>
<tr>
<td>11</td>
<td>SECURITY_STATUS_QUAL_RESUME</td>
</tr>
<tr>
<td>12</td>
<td>SECURITY_STATUS_QUAL_QUOTE_RESUME</td>
</tr>
<tr>
<td>13</td>
<td>SECURITY_STATUS_QUAL_TRADE_RESUME</td>
</tr>
<tr>
<td>14</td>
<td>SECURITY_STATUS_QUAL_RESUME_TIME</td>
</tr>
<tr>
<td>16</td>
<td>SECURITY_STATUS_QUAL_MKT_IMB_BUY</td>
</tr>
<tr>
<td>17</td>
<td>SECURITY_STATUS_QUAL_MKT_IMB_SELL</td>
</tr>
<tr>
<td>18</td>
<td>SECURITY_STATUS_QUAL_NO_MKT_IMB</td>
</tr>
<tr>
<td>19</td>
<td>SECURITY_STATUS_QUAL_MOC_IMB_BUY</td>
</tr>
<tr>
<td>20</td>
<td>SECURITY_STATUS_QUAL_MOC_IMB_SELL</td>
</tr>
<tr>
<td>21</td>
<td>SECURITY_STATUS_QUAL_NO_MOC_IMB</td>
</tr>
<tr>
<td>22</td>
<td>SECURITY_STATUS_QUAL_ORDER_IMB</td>
</tr>
<tr>
<td>23</td>
<td>SECURITY_STATUS_QUAL_ORDER_INF</td>
</tr>
<tr>
<td>24</td>
<td>SECURITY_STATUS_QUAL_ORDER_IMB_BUY</td>
</tr>
<tr>
<td>25</td>
<td>SECURITY_STATUS_QUAL_ORDER_IMB_SELL</td>
</tr>
<tr>
<td>26</td>
<td>SECURITY_STATUS_QUAL_ORDER_IMB_NONE</td>
</tr>
<tr>
<td>27</td>
<td>SECURITY_STATUS_QUAL_LOA_IMBALANCE_BUY</td>
</tr>
<tr>
<td>28</td>
<td>SECURITY_STATUS_QUAL_LOA_IMBALANCE_SELL</td>
</tr>
<tr>
<td>29</td>
<td>SECURITY_STATUS_QUAL_NO_LOA_IMBALANCE</td>
</tr>
<tr>
<td>30</td>
<td>SECURITY_STATUS_QUAL_ORDERS_ELIMINATED</td>
</tr>
<tr>
<td>31</td>
<td>SECURITY_STATUS_QUAL_RANGE_ID</td>
</tr>
<tr>
<td>32</td>
<td>SECURITY_STATUS_QUAL_ITS_PREOPEN</td>
</tr>
<tr>
<td>33</td>
<td>SECURITY_STATUS_QUAL_RESERVED</td>
</tr>
<tr>
<td>34</td>
<td>SECURITY_STATUS_QUAL_FROZEN</td>
</tr>
<tr>
<td>35</td>
<td>SECURITY_STATUS_QUAL_PREOPEN</td>
</tr>
<tr>
<td>36</td>
<td>SECURITY_STATUS_QUAL_THO_IMBALANCE_BUY</td>
</tr>
<tr>
<td>37</td>
<td>SECURITY_STATUS_QUAL_THO_IMBALANCE_SELL</td>
</tr>
<tr>
<td>38</td>
<td>SECURITY_STATUS_QUAL_NO_THO_IMBALANCE</td>
</tr>
</tbody>
</table>
Wombat::SECURITY_STATUS_QUAL_ADD_INFO = 41,
Wombat::SECURITY_STATUS_QUAL_IPO_IMBALANCE_BUY = 42,
Wombat::SECURITY_STATUS_QUAL_IPO_IMBALANCE_SELL = 43,
Wombat::SECURITY_STATUS_QUAL_NO_IPO_IMBALANCE = 44,
Wombat::SECURITY_STATUS_QUAL_EMC_IMB_BUY = 46,
Wombat::SECURITY_STATUS_QUAL_EMC_IMB_SELL = 47,
Wombat::SECURITY_STATUS_QUAL_EMC_IMB_NONE = 48,
Wombat::SECURITY_STATUS_QUAL_OPEN_DELAY = 51,
Wombat::SECURITY_STATUS_QUAL_NO_OPEN_NO_RESUME = 52,
Wombat::SECURITY_STATUS_QUAL_PRICE_IND = 53,
Wombat::SECURITY_STATUS_QUAL_EQUIPMENT = 54,
Wombat::SECURITY_STATUS_QUAL_FILINGS = 55,
Wombat::SECURITY_STATUS_QUAL_NEWS = 56,
Wombat::SECURITY_STATUS_QUAL_NEWS_DISSEM = 57,
Wombat::SECURITY_STATUS_QUAL_LISTING = 58,
Wombat::SECURITY_STATUS_OPERATION = 59,
Wombat::SECURITY_STATUS_QUAL_INFO = 60,
Wombat::SECURITY_STATUS_QUAL_SEC = 61,
Wombat::SECURITY_STATUS_QUAL_TIMES = 62,
Wombat::SECURITY_STATUS_QUAL_RELATED = 64,
Wombat::SECURITY_STATUS_QUAL_IPO = 65,
Wombat::SECURITY_STATUS_QUAL_PRE_CROSS = 66,
Wombat::SECURITY_STATUS_QUAL_CROSS = 67,
Wombat::SECURITY_STATUS_QUAL_RELEASED_FOR_QUOTATION = 68,
Wombat::SECURITY_STATUS_QUAL_IPO_WINDOW_EXT = 69,
Wombat::SECURITY_STATUS_QUAL_PYCLOSING = 70,
Wombat::SECURITY_STATUS_QUAL_AUCTION_EXTENSION = 71,
Wombat::SECURITY_STATUS_QUAL_VOLATILITY_AUCTION = 72,
Wombat::SECURITY_STATUS_QUAL_SECURITY_AUTHORISED = 73,
Wombat::SECURITY_STATUS_SECURITY_FORBIDDEN = 74,
Wombat::SECURITY_STATUS_QUAL_FAST_MARKET = 75,
Wombat::SECURITY_STATUS_QUAL_SLOW_MARKET = 76,
Wombat::SECURITY_STATUS_QUAL_SUB_PENNY_TRADING = 77,
Wombat::SECURITY_STATUS_QUAL_ORDER_INPUT = 78,
Wombat::SECURITY_STATUS_QUAL_PRE_ORDER_MATCHING = 79,
Wombat::SECURITY_STATUS_QUAL_ADJUST_ON = 88,
Wombat::SECURITY_STATUS_QUAL_LATE_TRADING = 89,
Wombat::SECURITY_STATUS_QUAL_ENQUIRE = 90,
Wombat::SECURITY_STATUS_QUAL_PRE_NIGHT_TRADING = 91,
Wombat::SECURITY_STATUS_QUAL_OPEN_NIGHT_TRADING = 92,
Wombat::SECURITY_STATUS_QUAL_SUB_HOLIDAY = 93,
Wombat::SECURITY_STATUS_RESERVED = 94,
Wombat::SECURITY_STATUS_RESERVED_RESERVED = 95,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED = 96,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED = 97,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 98,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 99,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 100,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 101,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 102,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 103,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 104,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 105,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 106,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 107,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 108,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 109,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 110,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 111,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 112,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 113,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 114,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 115,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 116,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 117,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 118,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 119,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 120,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 121,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 122,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 123,
Wombat::SECURITY_STATUS_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED_RESERVED = 124,

An enumeration representing the status of a security such as whether or not it is halted or closed for trading.
Functions

- MAMDAExpDLL const char * Wombat::toString (MamdaSecurityStatusQual securityStatusQual)
  
  Convert a MamdaSecurityStatusQual to an appropriate, displayable string.

- MAMDAExpDLL MamdaSecurityStatusQual Wombat::mamdaSecurityStatusQualFromString (const char *securityStatusQualStr)
  
  Convert a string representation of a security status to the enumeration.
#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>
#include <mama/marketdata.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaSubscription

  A MamdaSubscription is used to register interest in a particular symbol and source.
7.123 MamdaSymbolSourceEvent.h File Reference

#include <mamda/MamdaConfig.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaSymbolSourceEvent

  MamdaSymbolSourceEvent is an interface that provides access to a sourced symbol name.
7.124 MamdaSymbolSourceHandler.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaSymbolSourceEvent.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaSymbolSourceHandler

  MamdaSymbolSourceHandler is an interface for applications that want to have an easy way to handle newly sourced symbol events.
# include <mamda/MamdaConfig.h>
# include <mamda/MamdaBasicEvent.h>
# include <mamda/MamdaFieldState.h>

**Namespaces**

- namespace Wombat

**Classes**

- class Wombat::MamdaTradeCancelOrError
  
  *MamdaTradeCancelOrError is an interface that provides access to trade cancellation related fields.*
#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>
#include <mamda/MamdaCheckerHandler.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaTradeChecker

  MamdaTradeChecker is a class that provides trades sanity checking by periodically requesting snapshots of the trades from the publisher and comparing that with an trades being maintained in real time.
7.127 MamdaTradeClosing.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaTradeClosing

   MamdaTradeClosing is an interface that provides access to trade closing related fields.
7.128 MamdaTradeCorrection.h File Reference

#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaTradeCorrection

  *MamdaTradeCorrection* is an interface that provides access to trade correction related fields.
7.129 MamdaTradeDirection.h File Reference

#include <mamda/MamdaConfig.h>

Namespaces

• namespace Wombat

Enumerations

• enum Wombat::MamdaTradeDirection {
  Wombat::TRADE_DIR_ZERO,  Wombat::TRADE_DIR_PLUS,
  Wombat::TRADE_DIR_MINUS, Wombat::TRADE_DIR_ZERO_PLUS,
  Wombat::TRADE_DIR_ZERO_MINUS, Wombat::TRADE_DIR_NA,
  Wombat::TRADE_DIR_UNKNOWN = 99
}

An enumeration representing trade tick direction, relative to the previous "last" trade.

Functions

• MAMDAExpDLL const char * Wombat::toString (MamdaTradeDirection trade-Dir)
  
  Convert a MamdaTradeDirection to an appropriate, displayable string.

• MAMDAExpDLL MamdaTradeDirection Wombat::mamdaTradeDirection-FromString (const char *tradeDirStr)
  
  Convert a string representation of a trade direction to the enumeration.
7.130 MamdaTradeExecVenue.h File Reference

#include <mamda/MamdaConfig.h>

Namespaces

• namespace Wombat

Enumerations

• enum Wombat::MamdaTradeExecVenue {
    Wombat::TRADE_EXEC_VENUE_UNKNOWN, Wombat::TRADE_EXEC_VENUE_ON_EXCHANGE, Wombat::TRADE_EXEC_VENUE_ON_EXCHANGE_OFF_BOOK, Wombat::TRADE_EXEC_VENUE_OFF_EXCHANGE, Wombat::TRADE_EXEC_VENUE_SYSTEM_INTERNALISER, Wombat::TRADE_EXEC_VENUE_ON_EXCHANGE_DARK_BOOK, Wombat::TRADE_EXEC_VENUE_ON_EXCHANGE_ON_BOOK }

    An enumeration representing trade execution venue.

Functions

• MAMDAExpDLL const char * Wombat::toString (MamdaTradeExecVenue tradeExecVenue)
    Convert a MamdaTradeExecVenue to an appropriate, displayable string.

• MAMDAExpDLL MamdaTradeExecVenue Wombat::mamdaTradeExecVenueFromString (const char *tradeExecVenueStr)
    Convert a string representation of a trade execution venue to the enumeration.
7.131 MamdaTradeFields.h File Reference

#include <mamda/MamdaConfig.h>
#include <mama/mamacpp.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaTradeFields

Utility cache of MamaFieldDescriptors which are used internally by the API when accessing trade related fields from update messages.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaTradeGap

  *MamdaTradeGap is an interface that provides access to trade gap related fields.*
7.133 MamdaTradeHandler.h File Reference

```cpp
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaTradeListener.h>
#include <mama/mamacpp.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaTradeHandler

    _MamdaTradeHandler is an interface for applications that want to have an easy way to handle trade updates._
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaMsgListener.h>
#include <mamda/MamdaTradeRecap.h>
#include <mamda/MamdaTradeReport.h>
#include <mamda/MamdaTradeGap.h>
#include <mamda/MamdaTradeCancelOrError.h>
#include <mamda/MamdaTradeCorrection.h>
#include <mamda/MamdaTradeClosing.h>
#include <mamda/MamdaTradeDirection.h>
#include <mamda/MamdaTradeExecVenue.h>
#include <mamda/MamdaTradeOutOfSequence.h>
#include <mamda/MamdaTradePossiblyDuplicate.h>
#include <mamda/MamdaFieldState.h>
#include <mamda/MamdaTradeSide.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaTradeListener

  *MamdaTradeListener* is a class that specializes in handling trade updates.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaTradeOutOfSequence
  
  MamdaTradeOutOfSequence is an interface that provides access to fields related to trade updates which have been identified as being out of sequence with previous update (e.g.
# MamdaTradePossiblyDuplicate.h File Reference

```
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
```

## Namespaces

- namespace Wombat

## Classes

- class Wombat::MamdaTradePossiblyDuplicate
  
  `MamdaTradePossiblyDuplicate` is an interface that provides access to fields related to trade updates which have been identified as being possible duplicates of previous updates.
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicRecap.h>
#include <mamda/MamdaTradeDirection.h>
#include <mamda/MamdaTradeExecVenue.h>
#include <mamda/MamdaFieldState.h>

Namespaces

- namespace Wombat

Classes

- class Wombat::MamdaTradeRecap

  *MamdaTradeRecap is an interface that provides access to trade related fields.*
#include <mamda/MamdaConfig.h>
#include <mamda/MamdaBasicEvent.h>
#include <mamda/MamdaFieldState.h>

Namespaces

• namespace Wombat

Classes

• class Wombat::MamdaTradeReport

    *MamdaTradeReport is an interface that provides access to fields related to a trade report.*
Namespaces

- namespace Wombat

Enumerations

- enum Wombat::MamdaTradeSide { Wombat::TRADE_SIDE_UNKNOWN = 0,
  Wombat::TRADE_SIDE_BUY = 1, Wombat::TRADE_SIDE_SELL = 2 }

  An enumeration representing trade side.

Functions

- MAMDAExpDLL const char * Wombat::toString (MamdaTradeSide tradeSide)

  Convert a MamdaTradeSide to an appropriate, displayable string.

- MAMDAExpDLL MamdaTradeSide Wombat::mamdaTradeSideFromString (const char *tradeSideStr)

  Convert a string representation of a trade side to the enumeration.
#include <mamda/MamdaConfig.h>

Namespaces

- namespace Wombat

Enumerations

- enum Wombat::MamdaUncrossPriceInd { Wombat::UNCROSS_NONE, Wombat::UNCROSS_INDICATIVE, Wombat::UNCROSS_FIRM, Wombat::UNCROSS_INSUFFICIENT_VOL }

An enumeration representing the uncross price Ind.

Functions

- MAMDAExpDLL const char * Wombat::toString (MamdaUncrossPriceInd securityStatus)

  Convert a MamdaUncrossPriceInd to an appropriate, displayable string.

- MAMDAExpDLL MamdaUncrossPriceInd Wombat::mamdaUncrossPriceIndFromString (const char * uncrossPriceInd)

  Convert a string representation of a uncross price Ind to the enumeration.
#include <mamda/MamdaConfig.h>

**Namespaces**

- namespace *Wombat*

**Functions**

- MAMDAExpDLL const char *Wombat::getMamdaVersion (void)

*Get the version of Mamda.*