

TraDemGen

0.2.2

Generated by Doxygen 1.8.0

Fri Apr 27 2012 15:02:40

Contents

1	TraDemGen Documentation	2
1.1	Getting Started	2
1.2	TraDemGen at SourceForge	2
1.3	TraDemGen Development	2
1.4	External Libraries	2
1.5	Support TraDemGen	3
1.6	About TraDemGen	3
2	People	3
2.1	Project Admins (and Developers)	3
2.2	Retired Developers	3
2.3	Contributors	3
2.4	Distribution Maintainers	3
3	Coding Rules	4
3.1	Default Naming Rules for Variables	4
3.2	Default Naming Rules for Functions	4
3.3	Default Naming Rules for Classes and Structures	4
3.4	Default Naming Rules for Files	4
3.5	Default Functionality of Classes	4
4	Copyright and License	5
4.1	GNU LESSER GENERAL PUBLIC LICENSE	5
4.1.1	Version 2.1, February 1999	5
4.2	Preamble	5
4.3	TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION	6
4.3.1	NO WARRANTY	10
4.3.2	END OF TERMS AND CONDITIONS	10
4.4	How to Apply These Terms to Your New Programs	10
5	Documentation Rules	11
5.1	General Rules	11
5.2	File Header	12
5.3	Grouping Various Parts	12
6	Main features	12
6.1	Demand generation	12
6.2	Other features	13
7	Make a Difference	13

8	Make a new release	13
8.1	Introduction	13
8.2	Initialisation	14
8.3	Release branch maintenance	14
8.4	Commit and publish the release branch	14
8.5	Create distribution packages	14
8.6	Upload the HTML documentation to SourceForge	15
8.7	Generate the RPM packages	15
8.8	Update distributed change log	15
8.9	Create the binary package, including the documentation	15
8.10	Upload the files to SourceForge	16
8.11	Make a new post	16
8.12	Send an email on the announcement mailing-list	16
9	Installation	16
9.1	Table of Contents	16
9.2	Fedora/RedHat Linux distributions	16
9.3	TraDemGen Requirements	17
9.4	Basic Installation	17
9.5	Compilers and Options	18
9.6	Compiling For Multiple Architectures	18
9.7	Installation Names	19
9.8	Optional Features	19
9.9	Particular systems	20
9.10	Specifying the System Type	20
9.11	Sharing Defaults	21
9.12	Defining Variables	21
9.13	'cmake' Invocation	21
10	Linking with TraDemGen	25
10.1	Table of Contents	25
10.2	Introduction	25
10.3	Using the pkg-config command	25
10.4	Using the trademgen-config script	25
10.5	M4 macro for the GNU Autotools	26
10.6	Using TraDemGen with dynamic linking	26
11	Test Rules	26
11.1	The Test Source Files	26
11.2	The Reference File	27
11.3	Testing TraDemGen Library	27

12 Users Guide	27
12.1 Table of Contents	27
12.2 Introduction	27
12.3 Get Started	27
12.3.1 Get the TraDemGen library	27
12.3.2 Build the TraDemGen project	27
12.3.3 Build and Run the Tests	28
12.3.4 Install the TraDemGen Project (Binaries, Documentation)	28
12.4 Exploring the Predefined BOM Tree	28
12.4.1 Demand Stream Engine BOM Tree	28
12.5 Extending the BOM Tree	28
13 Supported Systems	28
13.1 Table of Contents	28
13.2 Introduction	28
13.3 TraDemGen 3.10.x	29
13.3.1 Linux Systems	29
13.3.2 Windows Systems	32
13.3.3 Unix Systems	35
14 TraDemGen Supported Systems (Previous Releases)	35
14.1 TraDemGen 3.9.1	35
14.2 TraDemGen 3.9.0	35
14.3 TraDemGen 3.8.1	35
15 Tutorials	35
15.1 Table of Contents	35
15.2 Introduction	36
15.2.1 Preparing the StdAir Project for Development	36
15.3 Build a Predefined BOM Tree	36
15.3.1 Instantiate the BOM Root Object	36
15.3.2 Instantiate the (Airline) Inventory Object	36
15.3.3 Link the Inventory Object with the BOM Root	36
15.3.4 Build Another Airline Inventory	37
15.3.5 Dump The BOM Tree Content	37
15.3.6 Result of the Tutorial Program	37
15.4 Extend the Pre-Defined BOM Tree	37
15.4.1 Extend an Airline Inventory Object	37
15.4.2 Build the Specific BOM Objects	37
15.4.3 Result of the Tutorial Program	38

16 Command-Line Test to Demonstrate How To Use TraDemGen elements	38
17 Directory Hierarchy	41
17.1 Directories	41
18 Namespace Index	42
18.1 Namespace List	42
19 Class Index	42
19.1 Class Hierarchy	42
20 Class Index	45
20.1 Class List	45
21 File Index	47
21.1 File List	47
22 Directory Documentation	49
22.1 trademgen/basic/ Directory Reference	49
22.2 trademgen/batches/ Directory Reference	50
22.3 trademgen/bom/ Directory Reference	50
22.4 trademgen/command/ Directory Reference	50
22.5 trademgen/config/ Directory Reference	50
22.6 trademgen/factory/ Directory Reference	50
22.7 trademgen/python/ Directory Reference	50
22.8 trademgen/ui/qt/ Directory Reference	51
22.9 trademgen/service/ Directory Reference	51
22.10test/ Directory Reference	51
22.11trademgen/ui/qt/trademgen/ Directory Reference	51
22.12trademgen/ Directory Reference	51
22.13test/trademgen/ Directory Reference	52
22.14trademgen/ui/ Directory Reference	52
23 Namespace Documentation	52
23.1 stdair Namespace Reference	52
23.1.1 Detailed Description	52
23.2 TRADEMGEN Namespace Reference	52
23.2.1 Typedef Documentation	55
23.2.2 Function Documentation	59
23.2.3 Variable Documentation	59
23.3 TRADEMGEN::DemandParserHelper Namespace Reference	60
23.3.1 Function Documentation	61
23.3.2 Variable Documentation	62

24 Class Documentation	63
24.1 BomAbstract Class Reference	63
24.2 TRADEMGEN::BomDisplay Class Reference	63
24.2.1 Detailed Description	64
24.2.2 Member Function Documentation	64
24.3 stdair::CategoricalAttribute< T > Struct Template Reference	64
24.3.1 Detailed Description	65
24.3.2 Member Typedef Documentation	65
24.3.3 Constructor & Destructor Documentation	65
24.3.4 Member Function Documentation	66
24.4 TRADEMGEN::CategoricalAttributeLite< T > Struct Template Reference	66
24.4.1 Detailed Description	67
24.4.2 Member Typedef Documentation	67
24.4.3 Constructor & Destructor Documentation	67
24.4.4 Member Function Documentation	68
24.5 CmdAbstract Class Reference	68
24.6 TRADEMGEN::ContinuousAttribute< T > Struct Template Reference	68
24.6.1 Detailed Description	69
24.6.2 Member Typedef Documentation	69
24.6.3 Constructor & Destructor Documentation	69
24.6.4 Member Function Documentation	70
24.7 TRADEMGEN::ContinuousAttributeLite< T > Struct Template Reference	70
24.7.1 Detailed Description	71
24.7.2 Member Typedef Documentation	71
24.7.3 Constructor & Destructor Documentation	71
24.7.4 Member Function Documentation	72
24.8 TRADEMGEN::DBManager Class Reference	72
24.8.1 Detailed Description	73
24.8.2 Member Function Documentation	73
24.9 TRADEMGEN::DBParams Struct Reference	73
24.9.1 Detailed Description	74
24.9.2 Constructor & Destructor Documentation	74
24.9.3 Member Function Documentation	74
24.10TRADEMGEN::DefaultMap Struct Reference	76
24.10.1 Detailed Description	76
24.10.2 Member Function Documentation	76
24.11TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT > Struct Template Reference	77
24.11.1 Detailed Description	78
24.11.2 Constructor & Destructor Documentation	79

24.11.3 Member Function Documentation	79
24.11.4 Member Data Documentation	79
24.12TRADEMGEN::DemandCharacteristics Struct Reference	82
24.12.1 Detailed Description	83
24.12.2 Constructor & Destructor Documentation	83
24.12.3 Member Function Documentation	84
24.12.4 Member Data Documentation	84
24.13TRADEMGEN::DemandDistribution Struct Reference	86
24.13.1 Detailed Description	86
24.13.2 Constructor & Destructor Documentation	86
24.13.3 Member Function Documentation	87
24.13.4 Member Data Documentation	87
24.14TRADEMGEN::DemandFileParser Class Reference	88
24.14.1 Detailed Description	88
24.14.2 Constructor & Destructor Documentation	88
24.14.3 Member Function Documentation	88
24.15DemandGenerationTestSuite Class Reference	89
24.15.1 Detailed Description	89
24.15.2 Constructor & Destructor Documentation	89
24.15.3 Member Function Documentation	89
24.15.4 Member Data Documentation	89
24.16TRADEMGEN::DemandInputFileNotFoundException Class Reference	89
24.16.1 Detailed Description	90
24.16.2 Constructor & Destructor Documentation	90
24.17TRADEMGEN::DemandManager Class Reference	90
24.17.1 Detailed Description	90
24.17.2 Friends And Related Function Documentation	91
24.18TRADEMGEN::DemandParserHelper::DemandParser Struct Reference	91
24.18.1 Detailed Description	91
24.18.2 Constructor & Destructor Documentation	92
24.18.3 Member Data Documentation	92
24.19TRADEMGEN::DemandParser Class Reference	92
24.19.1 Detailed Description	93
24.19.2 Member Function Documentation	93
24.20TRADEMGEN::DemandStream Class Reference	93
24.20.1 Detailed Description	95
24.20.2 Member Typedef Documentation	95
24.20.3 Constructor & Destructor Documentation	96
24.20.4 Member Function Documentation	96
24.20.5 Friends And Related Function Documentation	103

24.20.6 Member Data Documentation	103
24.21 TRADEMGEN::DemandStreamKey Struct Reference	105
24.21.1 Detailed Description	105
24.21.2 Constructor & Destructor Documentation	105
24.21.3 Member Function Documentation	106
24.22 TRADEMGEN::DemandStruct Struct Reference	107
24.22.1 Detailed Description	108
24.22.2 Constructor & Destructor Documentation	108
24.22.3 Member Function Documentation	108
24.22.4 Member Data Documentation	109
24.23 TRADEMGEN::DictionaryManager Class Reference	112
24.23.1 Detailed Description	112
24.23.2 Member Function Documentation	113
24.24 TRADEMGEN::DemandParserHelper::doEndDemand Struct Reference	113
24.24.1 Detailed Description	114
24.24.2 Constructor & Destructor Documentation	114
24.24.3 Member Function Documentation	114
24.24.4 Member Data Documentation	114
24.25 FacServiceAbstract Class Reference	115
24.26 TRADEMGEN::FacTRADEMGENSEerviceContext Class Reference	115
24.26.1 Detailed Description	116
24.26.2 Constructor & Destructor Documentation	116
24.26.3 Member Function Documentation	116
24.27 FileNotFoundException Class Reference	117
24.28 TRADEMGEN::FlagSaver Struct Reference	117
24.28.1 Detailed Description	117
24.28.2 Constructor & Destructor Documentation	117
24.29 grammar Class Reference	118
24.30 TRADEMGEN::IndexOutOfRangeException Class Reference	118
24.30.1 Detailed Description	118
24.30.2 Constructor & Destructor Documentation	118
24.31 KeyAbstract Class Reference	119
24.32 TRADEMGEN::DemandParserHelper::ParserSemanticAction Struct Reference	119
24.32.1 Detailed Description	120
24.32.2 Constructor & Destructor Documentation	120
24.32.3 Member Data Documentation	120
24.33 TRADEMGEN::RandomGenerationContext Struct Reference	120
24.33.1 Detailed Description	121
24.33.2 Constructor & Destructor Documentation	121
24.33.3 Member Function Documentation	121

24.34RootException Class Reference	122
24.35ServiceAbstract Class Reference	123
24.36TRADEMGEN::DemandParserHelper::storeChannelCode Struct Reference	123
24.36.1 Detailed Description	123
24.36.2 Constructor & Destructor Documentation	124
24.36.3 Member Function Documentation	124
24.36.4 Member Data Documentation	124
24.37TRADEMGEN::DemandParserHelper::storeChannelProbMass Struct Reference	124
24.37.1 Detailed Description	125
24.37.2 Constructor & Destructor Documentation	125
24.37.3 Member Function Documentation	125
24.37.4 Member Data Documentation	125
24.38TRADEMGEN::DemandParserHelper::storeDemandMean Struct Reference	126
24.38.1 Detailed Description	126
24.38.2 Constructor & Destructor Documentation	126
24.38.3 Member Function Documentation	126
24.38.4 Member Data Documentation	127
24.39TRADEMGEN::DemandParserHelper::storeDemandStdDev Struct Reference	127
24.39.1 Detailed Description	128
24.39.2 Constructor & Destructor Documentation	128
24.39.3 Member Function Documentation	128
24.39.4 Member Data Documentation	128
24.40TRADEMGEN::DemandParserHelper::storeDestination Struct Reference	129
24.40.1 Detailed Description	129
24.40.2 Constructor & Destructor Documentation	129
24.40.3 Member Function Documentation	129
24.40.4 Member Data Documentation	129
24.41TRADEMGEN::DemandParserHelper::storeDow Struct Reference	130
24.41.1 Detailed Description	130
24.41.2 Constructor & Destructor Documentation	130
24.41.3 Member Function Documentation	131
24.41.4 Member Data Documentation	131
24.42TRADEMGEN::DemandParserHelper::storeDTD Struct Reference	131
24.42.1 Detailed Description	132
24.42.2 Constructor & Destructor Documentation	132
24.42.3 Member Function Documentation	132
24.42.4 Member Data Documentation	132
24.43TRADEMGEN::DemandParserHelper::storeDTDProbMass Struct Reference	133
24.43.1 Detailed Description	133
24.43.2 Constructor & Destructor Documentation	133

24.43.3 Member Function Documentation	133
24.43.4 Member Data Documentation	134
24.44TRADEMGEN::DemandParserHelper::storeFFCode Struct Reference	134
24.44.1 Detailed Description	135
24.44.2 Constructor & Destructor Documentation	135
24.44.3 Member Function Documentation	135
24.44.4 Member Data Documentation	135
24.45TRADEMGEN::DemandParserHelper::storeFFProbMass Struct Reference	136
24.45.1 Detailed Description	136
24.45.2 Constructor & Destructor Documentation	136
24.45.3 Member Function Documentation	136
24.45.4 Member Data Documentation	136
24.46TRADEMGEN::DemandParserHelper::storeOrigin Struct Reference	137
24.46.1 Detailed Description	137
24.46.2 Constructor & Destructor Documentation	137
24.46.3 Member Function Documentation	138
24.46.4 Member Data Documentation	138
24.47TRADEMGEN::DemandParserHelper::storePosCode Struct Reference	138
24.47.1 Detailed Description	139
24.47.2 Constructor & Destructor Documentation	139
24.47.3 Member Function Documentation	139
24.47.4 Member Data Documentation	139
24.48TRADEMGEN::DemandParserHelper::storePosProbMass Struct Reference	140
24.48.1 Detailed Description	140
24.48.2 Constructor & Destructor Documentation	140
24.48.3 Member Function Documentation	140
24.48.4 Member Data Documentation	140
24.49TRADEMGEN::DemandParserHelper::storePrefCabin Struct Reference	141
24.49.1 Detailed Description	141
24.49.2 Constructor & Destructor Documentation	141
24.49.3 Member Function Documentation	142
24.49.4 Member Data Documentation	142
24.50TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd Struct Reference	142
24.50.1 Detailed Description	143
24.50.2 Constructor & Destructor Documentation	143
24.50.3 Member Function Documentation	143
24.50.4 Member Data Documentation	143
24.51TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart Struct Reference	144
24.51.1 Detailed Description	144
24.51.2 Constructor & Destructor Documentation	144

24.51.3 Member Function Documentation	144
24.51.4 Member Data Documentation	145
24.52TRADEMGEN::DemandParserHelper::storePrefDepTime Struct Reference	145
24.52.1 Detailed Description	146
24.52.2 Constructor & Destructor Documentation	146
24.52.3 Member Function Documentation	146
24.52.4 Member Data Documentation	146
24.53TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass Struct Reference	147
24.53.1 Detailed Description	147
24.53.2 Constructor & Destructor Documentation	147
24.53.3 Member Function Documentation	147
24.53.4 Member Data Documentation	147
24.54TRADEMGEN::DemandParserHelper::storeStayCode Struct Reference	148
24.54.1 Detailed Description	148
24.54.2 Constructor & Destructor Documentation	148
24.54.3 Member Function Documentation	149
24.54.4 Member Data Documentation	149
24.55TRADEMGEN::DemandParserHelper::storeStayProbMass Struct Reference	149
24.55.1 Detailed Description	150
24.55.2 Constructor & Destructor Documentation	150
24.55.3 Member Function Documentation	150
24.55.4 Member Data Documentation	150
24.56TRADEMGEN::DemandParserHelper::storeTimeValue Struct Reference	151
24.56.1 Detailed Description	151
24.56.2 Constructor & Destructor Documentation	151
24.56.3 Member Function Documentation	151
24.56.4 Member Data Documentation	151
24.57TRADEMGEN::DemandParserHelper::storeTimeValueProbMass Struct Reference	152
24.57.1 Detailed Description	152
24.57.2 Constructor & Destructor Documentation	152
24.57.3 Member Function Documentation	153
24.57.4 Member Data Documentation	153
24.58TRADEMGEN::DemandParserHelper::storeTripCode Struct Reference	153
24.58.1 Detailed Description	154
24.58.2 Constructor & Destructor Documentation	154
24.58.3 Member Function Documentation	154
24.58.4 Member Data Documentation	154
24.59TRADEMGEN::DemandParserHelper::storeTripProbMass Struct Reference	155
24.59.1 Detailed Description	155
24.59.2 Constructor & Destructor Documentation	155

24.59.3 Member Function Documentation	155
24.59.4 Member Data Documentation	156
24.60TRADEMGEN::DemandParserHelper::storeWTP Struct Reference	156
24.60.1 Detailed Description	157
24.60.2 Constructor & Destructor Documentation	157
24.60.3 Member Function Documentation	157
24.60.4 Member Data Documentation	157
24.61 StructAbstract Class Reference	158
24.62 TestFixture Class Reference	158
24.63TRADEMGEN::TRADEMGEN_Abstract Struct Reference	158
24.63.1 Detailed Description	159
24.63.2 Constructor & Destructor Documentation	159
24.63.3 Member Function Documentation	159
24.64TRADEMGEN::TRADEMGEN_Service Class Reference	160
24.64.1 Detailed Description	160
24.64.2 Constructor & Destructor Documentation	160
24.64.3 Member Function Documentation	161
24.65TRADEMGEN::TRADEMGEN_ServiceContext Class Reference	166
24.65.1 Detailed Description	167
24.65.2 Friends And Related Function Documentation	167
24.66TRADEMGEN::Trademgener Struct Reference	167
24.66.1 Detailed Description	167
24.66.2 Constructor & Destructor Documentation	168
24.66.3 Member Function Documentation	168
24.67TRADEMGEN::TrademgenGenerationException Class Reference	168
24.67.1 Detailed Description	169
24.67.2 Constructor & Destructor Documentation	169
25 File Documentation	169
25.1 doc/local/authors.doc File Reference	169
25.2 doc/local/codingrules.doc File Reference	169
25.3 doc/local/copyright.doc File Reference	169
25.4 doc/local/documentation.doc File Reference	169
25.5 doc/local/features.doc File Reference	169
25.6 doc/local/help_wanted.doc File Reference	169
25.7 doc/local/howto_release.doc File Reference	169
25.8 doc/local/index.doc File Reference	169
25.9 doc/local/installation.doc File Reference	169
25.10 doc/local/linking.doc File Reference	169
25.11 doc/local/test.doc File Reference	169

25.12doc/local/users_guide.doc File Reference	169
25.13doc/local/verification.doc File Reference	169
25.14doc/tutorial/tutorial.doc File Reference	169
25.15test/trademgen/DemandGenerationTestSuite.cpp File Reference	170
25.16DemandGenerationTestSuite.cpp	170
25.17test/trademgen/DemandGenerationTestSuite.hpp File Reference	173
25.17.1 Function Documentation	173
25.18DemandGenerationTestSuite.hpp	173
25.19test/trademgen/generateEvents.cpp File Reference	173
25.19.1 Function Documentation	174
25.20generateEvents.cpp	174
25.21trademgen/basic/BasConst.cpp File Reference	175
25.22BasConst.cpp	175
25.23trademgen/basic/BasConst_DemandGeneration.hpp File Reference	176
25.24BasConst_DemandGeneration.hpp	176
25.25trademgen/basic/BasConst_TRADEMGEN_Service.hpp File Reference	177
25.26BasConst_TRADEMGEN_Service.hpp	177
25.27trademgen/basic/BasParserTypes.hpp File Reference	177
25.28BasParserTypes.hpp	179
25.29trademgen/basic/CategoricalAttribute.hpp File Reference	180
25.30CategoricalAttribute.hpp	180
25.31trademgen/basic/CategoricalAttributeLite.hpp File Reference	182
25.32CategoricalAttributeLite.hpp	182
25.33trademgen/basic/ContinuousAttribute.hpp File Reference	184
25.34ContinuousAttribute.hpp	184
25.35trademgen/basic/ContinuousAttributeLite.hpp File Reference	186
25.36ContinuousAttributeLite.hpp	186
25.37trademgen/basic/DemandCharacteristics.cpp File Reference	189
25.38DemandCharacteristics.cpp	189
25.39trademgen/basic/DemandCharacteristics.hpp File Reference	191
25.40DemandCharacteristics.hpp	191
25.41trademgen/basic/DemandCharacteristicsTypes.hpp File Reference	192
25.42DemandCharacteristicsTypes.hpp	193
25.43trademgen/basic/DemandDistribution.cpp File Reference	194
25.44DemandDistribution.cpp	194
25.45trademgen/basic/DemandDistribution.hpp File Reference	195
25.46DemandDistribution.hpp	195
25.47trademgen/basic/DictionaryManager.cpp File Reference	196
25.48DictionaryManager.cpp	196
25.49trademgen/basic/DictionaryManager.hpp File Reference	196

25.50DictionaryManager.hpp	197
25.51trademgen/basic/RandomGenerationContext.cpp File Reference	197
25.52RandomGenerationContext.cpp	197
25.53trademgen/basic/RandomGenerationContext.hpp File Reference	198
25.54RandomGenerationContext.hpp	198
25.55trademgen/batches/trademgen.cpp File Reference	199
25.55.1 Typedef Documentation	200
25.55.2 Function Documentation	200
25.55.3 Variable Documentation	201
25.56trademgen.cpp	202
25.57trademgen/ui/qt/trademgen/trademgen.cpp File Reference	208
25.58trademgen.cpp	208
25.59trademgen/batches/trademgen_with_db.cpp File Reference	208
25.59.1 Typedef Documentation	209
25.59.2 Function Documentation	209
25.59.3 Variable Documentation	210
25.60trademgen_with_db.cpp	211
25.61trademgen/bom/BomDisplay.cpp File Reference	215
25.62BomDisplay.cpp	215
25.63trademgen/bom/BomDisplay.hpp File Reference	216
25.64BomDisplay.hpp	217
25.65trademgen/bom/DemandStream.cpp File Reference	217
25.66DemandStream.cpp	217
25.67trademgen/bom/DemandStream.hpp File Reference	224
25.68DemandStream.hpp	225
25.69trademgen/bom/DemandStreamKey.cpp File Reference	228
25.70DemandStreamKey.cpp	228
25.71trademgen/bom/DemandStreamKey.hpp File Reference	229
25.72DemandStreamKey.hpp	229
25.73trademgen/bom/DemandStreamTypes.hpp File Reference	230
25.74DemandStreamTypes.hpp	230
25.75trademgen/bom/DemandStruct.cpp File Reference	231
25.76DemandStruct.cpp	231
25.77trademgen/bom/DemandStruct.hpp File Reference	233
25.78DemandStruct.hpp	233
25.79trademgen/command/DBManager.cpp File Reference	234
25.80DBManager.cpp	235
25.81trademgen/command/DBManager.hpp File Reference	237
25.82DBManager.hpp	237
25.83trademgen/command/DemandManager.cpp File Reference	238

25.84DemandManager.cpp	238
25.85trademgen/command/DemandManager.hpp File Reference	248
25.86DemandManager.hpp	248
25.87trademgen/command/DemandParser.cpp File Reference	250
25.88DemandParser.cpp	250
25.89trademgen/command/DemandParser.hpp File Reference	251
25.90DemandParser.hpp	251
25.91trademgen/command/DemandParserHelper.cpp File Reference	251
25.92DemandParserHelper.cpp	252
25.93trademgen/command/DemandParserHelper.hpp File Reference	262
25.94DemandParserHelper.hpp	263
25.95trademgen/config/trademgen-paths.hpp File Reference	266
25.95.1 Define Documentation	266
25.96trademgen-paths.hpp	268
25.97trademgen/DBParams.hpp File Reference	268
25.98DBParams.hpp	268
25.99trademgen/factory/FacTRADEMGENSEerviceContext.cpp File Reference	270
25.100FacTRADEMGENSEerviceContext.cpp	270
25.101trademgen/factory/FacTRADEMGENSEerviceContext.hpp File Reference	271
25.102FacTRADEMGENSEerviceContext.hpp	271
25.103trademgen/python/pytrademgen.cpp File Reference	272
25.103.1 Function Documentation	272
25.104pytrademgen.cpp	272
25.105trademgen/service/TRADEMGEN_Service.cpp File Reference	274
25.106TRADEMGEN_Service.cpp	275
25.107trademgen/service/TRADEMGEN_ServiceContext.cpp File Reference	282
25.108TRADEMGEN_ServiceContext.cpp	282
25.109trademgen/service/TRADEMGEN_ServiceContext.hpp File Reference	283
25.110TRADEMGEN_ServiceContext.hpp	283
25.111trademgen/TRADEMGEN_Abstract.hpp File Reference	285
25.111.1 Function Documentation	285
25.112TRADEMGEN_Abstract.hpp	285
25.113trademgen/TRADEMGEN_Exceptions.hpp File Reference	286
25.114TRADEMGEN_Exceptions.hpp	287
25.115trademgen/TRADEMGEN_Service.hpp File Reference	287
25.116TRADEMGEN_Service.hpp	287
25.117trademgen/TRADEMGEN_Types.hpp File Reference	289
25.118TRADEMGEN_Types.hpp	289
25.119trademgen/ui/qt/trademgen/main.cpp File Reference	290
25.119.1 Function Documentation	290

25.120main.cpp	290
--	-----

1 TraDemGen Documentation

1.1 Getting Started

- [Main features](#)
- [Installation](#)
- [Linking with TraDemGen](#)
- [Users Guide](#)
- [Tutorials](#)
- [Copyright and License](#)
- [Make a Difference](#)
- [Make a new release](#)
- [People](#)

1.2 TraDemGen at SourceForge

- [Project page](#)
- [Download TraDemGen](#)
- [Open a ticket for a bug or feature](#)
- [Mailing lists](#)
- [Forums](#)
 - [Discuss about Development issues](#)
 - [Ask for Help](#)
 - [Discuss TraDemGen](#)

1.3 TraDemGen Development

- [Git Repository](#) (Subversion is deprecated)
- [Coding Rules](#)
- [Documentation Rules](#)
- [Test Rules](#)

1.4 External Libraries

- [Boost](#) (C++ STL extensions)
- [Python](#)
- [MySQL client](#)
- [SOI](#) (C++ DB API)

1.5 Support TraDemGen

1.6 About TraDemGen

TraDemGen aims at providing a clean API, and the corresponding C++ implementation, able to generate demand for travel solutions (e.g., from JFK to PEK on 25-05-2009) according to characteristics (e.g., Willingness-To-Pay, preferred airline, etc). TraDemGen mainly targets simulation purposes. [N](#)

TraDemGen makes an extensive use of existing open-source libraries for increased functionality, speed and accuracy. In particular the [Boost](#) (*C++ Standard Extensions*) library is used.

The TraDemGen library originates from the department of Operational Research and Innovation at [Amadeus](#), Sophia Antipolis, France. TraDemGen is released under the terms of the [GNU Lesser General Public License](#) (LGPLv2.1) for you to enjoy.

TraDemGen should work on [GNU/Linux](#), [Sun Solaris](#), Microsoft Windows (with [Cygwin](#), [MinGW/MSYS](#), or [Microsoft Visual C++ .NET](#)) and [Mac OS X](#) operating systems.

Note

(N) - The TraDemGen library is **NOT** intended, in any way, to be used by airlines for production systems. If you want to report issue, bug or feature request, or if you just want to give feedback, have a look on the right-hand side of this page for the preferred reporting methods. In any case, please do not contact Amadeus directly for any matter related to TraDemGen.

2 People

2.1 Project Admins (and Developers)

- Anh Quan Nguyen quannaus@users.sourceforge.net ([N](#))
- Denis Arnaud denis_arnaud@users.sourceforge.net ([N](#))
- Gabrielle Sabatier gsabatier@users.sourceforge.net ([N](#))

2.2 Retired Developers

- Mehdi Ayouni mehdi.ayouni@gmail.com
- Son Nguyen Kim snguyenkim@users.sourceforge.net ([N](#))

2.3 Contributors

- Emmanuel Bastien ebastien@users.sourceforge.net ([N](#))

2.4 Distribution Maintainers

- [Fedora/RedHat](#): Denis Arnaud denis_arnaud@users.sourceforge.net ([N](#))
- [Debian](#): Emmanuel Bastien ebastien@users.sourceforge.net ([N](#))

Note

(N) - [Amadeus](#) employees.

3 Coding Rules

In the following sections we describe the naming conventions which are used for files, classes, structures, local variables, and global variables.

3.1 Default Naming Rules for Variables

Variables names follow Java naming conventions. Examples:

- `lNumberOfPassengers`
- `lSeatAvailability`

3.2 Default Naming Rules for Functions

Function names follow Java naming conventions. Example:

- `int myFunctionName (const int& a, int b)`

3.3 Default Naming Rules for Classes and Structures

Each new word in a class or structure name should always start with a capital letter and the words should be separated with an under-score. Abbreviations are written with capital letters. Examples:

- `MyClassName`
- `MyStructName`

3.4 Default Naming Rules for Files

Files are named after the C++ class names.

Source files are named using `.cpp` suffix, whereas header files end with `.hpp` extension. Examples:

- `FlightDate.hpp`
- `SegmentDate.cpp`

3.5 Default Functionality of Classes

All classes that are configured by input parameters should include:

- default empty constructor
- one or more additional constructor(s) that takes input parameters and initializes the class instance
- setup function, preferably named `'setup'` or `'set_parameters'`

Explicit destructor functions are not required, unless they are needed. It shall not be possible to use any of the other member functions unless the class has been properly initiated with the input parameters.

4 Copyright and License

4.1 GNU LESSER GENERAL PUBLIC LICENSE

4.1.1 Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

4.2 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

4.3 TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

1. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no

charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

1. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

1. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

1. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is

especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

1. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

1. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

(a) You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

1. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.
1. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.
1. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

1. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
1. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

1. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

4.3.1 NO WARRANTY

1. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
1. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

4.3.2 END OF TERMS AND CONDITIONS

4.4 How to Apply These Terms to Your New Programs

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

```
This library is free software; you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the License, or (at your option) any later version.
```

```
This library is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Lesser General Public License for more details.
```

```
You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
```

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

[Source](#)

5 Documentation Rules

5.1 General Rules

All classes in TraDemGen should be properly documented with Doxygen comments in include (.hpp) files. Source (.cpp) files should be documented according to a normal standard for well documented C++ code.

An example of how the interface of a class shall be documented in TraDemGen is shown here:

```

/*!
 * \brief Brief description of MyClass here
 *
 * Detailed description of MyClass here. With example code if needed.
 */
class MyClass {
public:
    //! Default constructor
    MyClass(void) { setup_done = false; }

    /*!
     * \brief Constructor that initializes the class with parameters
     *
     * Detailed description of the constructor here if needed
     *
     * \param[in] param1 Description of \a param1 here
     * \param[in] param2 Description of \a param2 here
     */
    MyClass(TYPE1 param1, TYPE2 param2) { setup(param1, param2); }

    /*!
     * \brief Setup function for MyClass
     *
     * Detailed description of the setup function here if needed
     *
     * \param[in] param1 Description of \a param1 here
     * \param[in] param2 Description of \a param2 here
     */
    void setup(TYPE1 param1, TYPE2 param2);

    /*!
     * \brief Brief description of memberFunction1
     *
     * Detailed description of memberFunction1 here if needed
     *
     * \param[in] param1 Description of \a param1 here
     * \param[in] param2 Description of \a param2 here
     * \param[in,out] param3 Description of \a param3 here
     * \return Description of the return value here
     */
    TYPE4 memberFunction1(TYPE1 param1, TYPE2 param2, TYPE3 &param3);

private:
    bool _setupDone;          /*!< Variable that checks if the class is properly
                               initialized with parameters */
    TYPE1 _privateVariable1; /*!< Short description of _privateVariable1 here
    TYPE2 _privateVariable2; /*!< Short description of _privateVariable2 here
};

```

5.2 File Header

All files should start with the following header, which include Doxygen's `\file`, `\brief` and `\author` tags, `$Date$` and `$Revisions$` CVS tags, and a common copyright note:

```

/ *!
 * \file
 * \brief Brief description of the file here
 * \author Names of the authors who contributed to this code
 * \date Date
 *
 * Detailed description of the file here if needed.
 *
 * -----
 *
 * TraDemGen - C++ Simulated Revenue Accounting (RAC) System Library
 *
 * Copyright (C) 2009-2011 (\see authors file for a list of contributors)
 *
 * \see copyright file for license information
 *
 * -----
 */

```

5.3 Grouping Various Parts

All functions must be added to a Doxygen group in order to appear in the documentation. The following code example defines the group `'my_group'`:

```

/ *!
 * \defgroup my_group Brief description of the group here
 *
 * Detailed description of the group here
 */

```

The following example shows how to document the function `myFunction` and how to add it to the group `my_group`:

```

/ *!
 * \brief Brief description of myFunction here
 * \ingroup my_group
 *
 * Detailed description of myFunction here
 *
 * \param[in] param1 Description of \a param1 here
 * \param[in] param2 Description of \a param2 here
 * \return Description of the return value here
 */
TYPE3 myFunction(TYPE1 param1, TYPE2 &param2);

```

6 Main features

A short list of the main features of TraDemgEn is given below sorted in different categories. Many more features and functions exist and for these we refer to the reference documentation.

6.1 Demand generation

The demand can be generated thanks to two relatively advanced pieces of algorithm, both following a sequential principle. That is, the **events** (booking requests) are generated one after the other, sequentially, rather than being generated all at once at the beginning of the process (e.g., a simulation).

The two sequential methods are:

- 'Intuitive' method. The booking period is sliced in `intervals`, where the arrival rate of events (booking requests) is known for each of those intervals, say λ_i . The inter-arrival process then follows an `exponential law`. That is, the final number of booking requests follows a `Non homogeneous Poisson distribution`. With that method, the `variance` of that `distribution` is therefore equal to the `mean`.
- 'Advanced' method. The process uses `order statistics` in order to mimic the behaviour of `uniform distributions` projected onto the know arrival pattern of events. With that method, the final number of booking requests is first drawn, following any probability distribution (e.g., `normal`, `Gamma`, `Beta` or even Weibull law) with any required standard deviation. Then, each booking request is drawn in sequence:
 - according to a mere `uniform distribution`,
 - and projected onto the known booking arrival pattern.

6.2 Other features

- CSV input file parsing
- Memory handling

7 Make a Difference

Do not ask what TraDemGen can do for you. Ask what you can do for TraDemGen.

You can help us to develop the TraDemGen library. There are always a lot of things you can do:

- Start using TraDemGen
- Tell your friends about TraDemGen and help them to get started using it
- If you find a bug, report it to us. Without your help we can never hope to produce a bug free code.
- Help us to improve the documentation by providing information about documentation bugs
- Answer support requests in the TraDemGen discussion forums on SourceForge. If you know the answer to a question, help others to overcome their TraDemGen problems.
- Help us to improve our algorithms. If you know of a better way (e.g. that is faster or requires less memory) to implement some of our algorithms, then let us know.
- Help us to port TraDemGen to new platforms. If you manage to compile TraDemGen on a new platform, then tell us how you did it.
- Send us your code. If you have a good TraDemGen compatible code, which you can release under the LGPL, and you think it should be included in TraDemGen, then send it to us.
- Become an TraDemGen developer. Send us an e-mail and tell what you can do for TraDemGen.

8 Make a new release

8.1 Introduction

This document describes briefly the recommended procedure of releasing a new version of TraDemGen using a Linux development machine and the SourceForge project site.

The following steps are required to make a release of the distribution package.

8.2 Initialisation

Clone locally the full [Git project](#):

```
cd ~
mkdir -p dev/sim
cd ~/dev/sim
git clone git://trademgen.git.sourceforge.net/gitroot/trademgen/trademgen trademgengit
cd trademgengit
git checkout trunk
```

8.3 Release branch maintenance

Switch to the release branch, on your local clone, and merge the latest updates from the trunk. Decide about the new version to be released.

```
cd ~/dev/sim/trademgengit
git checkout releases
git merge trunk
```

Update the version in the various build system files, replacing the old version numbers by the correct ones:

```
vi CMakeLists.txt
vi autogen.sh
vi README
```

Update the version, add some news in the NEWS file, add a change-log in the ChangeLog file and in the RPM specification files:

```
vi NEWS
vi ChangeLog
vi trademgen.spec
```

8.4 Commit and publish the release branch

Commit the new release:

```
cd ~/dev/sim/trademgengit
git add -A
git commit -m "[Release 0.5.0] Release of the 0.5.0 version of TraDemGen."
git push
```

8.5 Create distribution packages

Create the distribution packages using the following command:

```
cd ~/dev/sim/trademgengit
git checkout releases
rm -rf build && mkdir -p build
cd build
export INSTALL_BASEDIR=/home/user/dev/deliveries
export LIBSUFFIX_4_CMAKE="-DLIB_SUFFIX=64"
cmake -DCMAKE_INSTALL_PREFIX=${INSTALL_BASEDIR}/trademgen-0.5.0 \
  -DWITH_STDAIR_PREFIX=${INSTALL_BASEDIR}/stdair-stable \
  -DWITH_AIRAC_PREFIX=${INSTALL_BASEDIR}/airsched-stable \
  -DWITH_AIRAC_PREFIX=${INSTALL_BASEDIR}/airrac-stable \
  -DWITH_RMOL_PREFIX=${INSTALL_BASEDIR}/rmol-stable \
  -DWITH_RMOL_PREFIX=${INSTALL_BASEDIR}/airinv-stable \
  -DWITH_RMOL_PREFIX=${INSTALL_BASEDIR}/simfqt-stable \
  -DCMAKE_BUILD_TYPE:STRING=Debug -DINSTALL_DOC:BOOL=ON \
  ${LIBSUFFIX_4_CMAKE} ..
make check && make dist
make install
```

This will configure, compile and check the package. The output packages will be named, for instance, trademgen-0.5.0.tar.gz and trademgen-0.5.0.tar.bz2.

8.6 Upload the HTML documentation to SourceForge

In order to update the Web site files, either:

- **synchronise them with rsync and SSH:** Upload the just generated HTML (and PDF) documentation onto the **SourceForge Web site**.

```
cd ~/dev/sim/trademgengit/build
git checkout releases
rsync -aiv ${INSTALL_BASEDIR}/trademgen-0.5.0/share/doc/trademgen-0.5.0/html/ \
  your_sf_user,trademgen@web.sourceforge.net:htdocs/
```

where `-aiv` options mean:

- `-a`: archive/mirror mode; equals `-rlptgoD` (no `-H`, `-A`, `-X`)
- `-v`: increase verbosity
- `-i`: output a change-summary for all updates
- Note the trailing slashes (/) at the end of both the source and target directories. It means that the content of the source directory (`doc/html`), rather than the directory itself, has to be copied into the content of the target directory.

- or use the **SourceForge Shell service**.

8.7 Generate the RPM packages

Optionally, generate the RPM package (for instance, for **Fedora/RedHat**):

```
cd ~/dev/sim/trademgengit/build
git checkout releases
make dist
```

To perform this step, `rpm-build`, `rpmlint` and `rpmdevtools` have to be available on the system.

```
cp ../trademgen.spec ~/dev/packages/SPECS \
  && cp trademgen-0.5.0.tar.bz2 ~/dev/packages/SOURCES
cd ~/dev/packages/SPECS
rpmbuild -ba trademgen.spec
cd ~/dev/packages
rpmlint -i SPECS/trademgen.spec SRPMS/trademgen-0.5.0-1.fc16.src.rpm \
  RPMS/noarch/trademgen-* RPMS/i686/trademgen-*
```

8.8 Update distributed change log

Update the `NEWS` and `ChangeLog` files with appropriate information, including what has changed since the previous release. Then commit and push the changes into the **TraDemGen's Git repository**.

8.9 Create the binary package, including the documentation

Create the binary package, which includes HTML and PDF documentation, using the following command:

```
cd ~/dev/sim/trademgengit/build
git checkout releases
make package
```

The output binary package will be named, for instance, `trademgen-0.5.0-Linux.tar.bz2`. That package contains both the HTML and PDF documentation. The binary package contains also the executables and shared libraries, as well as C++ header files, but all of those do not interest us for now.

8.10 Upload the files to SourceForge

Upload the distribution and documentation packages to the SourceForge server. Check [SourceForge help page on uploading software](#).

8.11 Make a new post

- submit a new entry in the [SourceForge project-related news feed](#)
- make a new post on the [SourceForge hosted WordPress blog](#)
- and update, if necessary, [Trac tickets](#).

8.12 Send an email on the announcement mailing-list

Finally, you should send an announcement to trademgen-announce@lists.sourceforge.net (see <https://lists.sourceforge.net/lists/listinfo/trademgen-announce> for the archives)

9 Installation

9.1 Table of Contents

- [Fedora/RedHat Linux distributions](#)
- [TraDemGen Requirements](#)
- [Basic Installation](#)
- [Compilers and Options](#)
- [Compiling For Multiple Architectures](#)
- [Installation Names](#)
- [Optional Features](#)
- [Particular systems](#)
- [Specifying the System Type](#)
- [Sharing Defaults](#)
- [Defining Variables](#)
- ['cmake' Invocation](#)

9.2 Fedora/RedHat Linux distributions

Note that on [Fedora/RedHat](#) Linux distributions, RPM packages are available and can be installed with your usual package manager. For instance:

```
yum -y install trademgen-devel trademgen-doc
```

RPM packages can also be available on the [SourceForge download site](#).

9.3 TraDemGen Requirements

TraDemGen should compile without errors or warnings on most GNU/Linux systems, on UNIX systems like Solaris SunOS, and on POSIX based environments for Microsoft Windows like Cygwin or MinGW with MSYS. It can be also built on Microsoft Windows NT/2000/XP/Vista/7 using Microsoft's Visual C++ .NET, but our support for this compiler is limited. For GNU/Linux, SunOS, Cygwin and MinGW we assume that you have at least the following GNU software installed on your computer:

- GNU Autotools:
 - `autoconf`,
 - `automake`,
 - `libtool`,
 - `make`, version 3.72.1 or later (check version with `'make --version'`)
- `GCC` - GNU C++ Compiler (g++), version 4.3.x or later (check version with `'gcc --version'`)
- `Boost` - C++ STL extensions, version 1.35 or later (check version with `'grep "define BOOST_LIB_VERSION" /usr/include/boost/version.hpp'`)
- `MySQL` - Database client libraries, version 5.0 or later (check version with `'mysql --version'`)
- `SOCI` - C++ database client library wrapper, version 3.0.0 or later (check version with `'soci-config --version'`)

Optionally, you might need a few additional programs: `Doxygen`, `LaTeX`, `Dvips` and `Ghostscript`, to generate the HTML and PDF documentation.

We strongly recommend that you use recent stable releases of the GCC, if possible. We do not actively work on supporting older versions of the GCC, and they may therefore (without prior notice) become unsupported in future releases of TraDemGen.

9.4 Basic Installation

Briefly, the shell commands `./cmake .. && make install` should configure, build and install this package. The following more-detailed instructions are generic; see the `'README'` file for instructions specific to this package. Some packages provide this `'INSTALL'` file but do not implement all of the features documented below. The lack of an optional feature in a given package is not necessarily a bug. More recommendations for GNU packages can be found in the info page corresponding to "Makefile Conventions: (standards)Makefile Conventions".

The `'cmake'` shell script attempts to guess correct values for various system-dependent variables used during compilation. It uses those values to create a `'Makefile'` in each directory of the package. It may also create one or more `'h'` files containing system-dependent definitions. Finally, it creates a `'CMakeCache.txt'` cache file that you can refer to in the future to recreate the current configuration, and files `'CMakeFiles'` containing compiler output (useful mainly for debugging `'cmake'`).

It can also use an optional file (typically called `'config.cache'` and enabled with `'--cache-file=config.cache'` or simply `'-C'`) that saves the results of its tests to speed up reconfiguring. Caching is disabled by default to prevent problems with accidental use of stale cache files.

If you need to do unusual things to compile the package, please try to figure out how `'configure'` could check whether to do them, and mail diffs or instructions to the address given in the `'README'` so they can be considered for the next release. If you are using the cache, and at some point `'config.cache'` contains results you don't want to keep, you may remove or edit it.

The file `'CMakeLists.txt'` is used to create the `'Makefile'` files.

The simplest way to compile this package is:

1. `'cd'` to the directory containing the package's source code and type `./cmake ..` to configure the package for your system. Running `'cmake'` is generally fast. While running, it prints some messages telling which features it is checking for.

2. Type `'make'` to compile the package.
3. Optionally, type `'make check'` to run any self-tests that come with the package, generally using the just-built uninstalled binaries.
4. Type `'make install'` to install the programs and any data files and documentation. When installing into a prefix owned by root, it is recommended that the package be configured and built as a regular user, and only the `'make install'` phase executed with root privileges.
5. You can remove the program binaries and object files from the source code directory by typing `'make clean'`. To also remove the files that `'configure'` created (so you can compile the package for a different kind of computer), type `'make distclean'`. There is also a `'make maintainer-clean'` target, but that is intended mainly for the package's developers. If you use it, you may have to get all sorts of other programs in order to regenerate files that came with the distribution.
6. Often, you can also type `'make uninstall'` to remove the installed files again. In practice, not all packages have tested that uninstallation works correctly, even though it is required by the GNU Coding Standards.

9.5 Compilers and Options

Some systems require unusual options for compilation or linking that the `'cmake'` script does not know about. Run `'./cmake --help'` for details on some of the pertinent environment variables.

You can give `'cmake'` initial values for configuration parameters by setting variables in the command line or in the environment. Here is an example:

```
./cmake CC=c99 CFLAGS=-g LIBS=-lposix
```

See also

[Defining Variables](#) for more details.

9.6 Compiling For Multiple Architectures

You can compile the package for more than one kind of computer at the same time, by placing the object files for each architecture in their own directory. To do this, you can use GNU `'make'`. `'cd'` to the directory where you want the object files and executables to go and run the `'configure'` script. `'configure'` automatically checks for the source code in the directory that `'configure'` is in and in `'..'`. This is known as a "VPATH" build.

With a non-GNU `'make'`, it is safer to compile the package for one architecture at a time in the source code directory. After you have installed the package for one architecture, use `'make distclean'` before reconfiguring for another architecture.

On MacOS X 10.5 and later systems, you can create libraries and executables that work on multiple system types--known as "fat" or "universal" binaries--by specifying multiple `'-arch'` options to the compiler but only a single `'-arch'` option to the preprocessor. Like this:

```
./configure CC="gcc -arch i386 -arch x86_64 -arch ppc -arch ppc64" \
           CXX="g++ -arch i386 -arch x86_64 -arch ppc -arch ppc64" \
           CPP="gcc -E" CXXCPP="g++ -E"
```


This is not guaranteed to produce working output in all cases, you may have to build one architecture at a time and combine the results using the 'lipo' tool if you have problems.

9.7 Installation Names

By default, 'make install' installs the package's commands under '/usr/local/bin', include files under '/usr/local/include', etc. You can specify an installation prefix other than '/usr/local' by giving 'configure' the option '--prefix=PREFIX', where PREFIX must be an absolute file name.

You can specify separate installation prefixes for architecture-specific files and architecture-independent files. If you pass the option '--exec-prefix=PREFIX' to 'configure', the package uses PREFIX as the prefix for installing programs and libraries. Documentation and other data files still use the regular prefix.

In addition, if you use an unusual directory layout you can give options like '--bindir=DIR' to specify different values for particular kinds of files. Run 'configure --help' for a list of the directories you can set and what kinds of files go in them. In general, the default for these options is expressed in terms of '\${prefix}', so that specifying just '--prefix' will affect all of the other directory specifications that were not explicitly provided.

The most portable way to affect installation locations is to pass the correct locations to 'configure'; however, many packages provide one or both of the following shortcuts of passing variable assignments to the 'make install' command line to change installation locations without having to reconfigure or recompile.

The first method involves providing an override variable for each affected directory. For example, 'make install prefix=/alternate/directory' will choose an alternate location for all directory configuration variables that were expressed in terms of '\${prefix}'. Any directories that were specified during 'configure', but not in terms of '\${prefix}', must each be overridden at install time for the entire installation to be relocated. The approach of makefile variable overrides for each directory variable is required by the GNU Coding Standards, and ideally causes no recompilation. However, some platforms have known limitations with the semantics of shared libraries that end up requiring recompilation when using this method, particularly noticeable in packages that use GNU Libtool.

The second method involves providing the 'DESTDIR' variable. For example, 'make install DESTDIR=/alternate/directory' will prepend '/alternate/directory' before all installation names. The approach of 'DESTDIR' overrides is not required by the GNU Coding Standards, and does not work on platforms that have drive letters. On the other hand, it does better at avoiding recompilation issues, and works well even when some directory options were not specified in terms of '\${prefix}' at 'configure' time.

9.8 Optional Features

If the package supports it, you can cause programs to be installed with an extra prefix or suffix on their names by giving 'cmake' the option '--program-prefix=PREFIX' or '--program-suffix=SUFFIX'.

Some packages pay attention to '--enable-FEATURE' options to 'configure', where FEATURE indicates an optional part of the package. They may also pay attention to '--with-PACKAGE' options, where PACKAGE is something like

'gnu-as' or 'x' (for the X Window System). The 'README' should mention any '--enable-' and '--with-' options that the package recognizes.

For packages that use the X Window System, 'configure' can usually find the X include and library files automatically, but if it doesn't, you can use the 'configure' options '--x-includes=DIR' and '--x-libraries=DIR' to specify their locations.

Some packages offer the ability to configure how verbose the execution of 'make' will be. For these packages, running './configure --enable-silent-rules' sets the default to minimal output, which can be overridden with 'make V=1'; while running './configure --disable-silent-rules' sets the default to verbose, which can be overridden with 'make V=0'.

9.9 Particular systems

On HP-UX, the default C compiler is not ANSI C compatible. If GNU CC is not installed, it is recommended to use the following options in order to use an ANSI C compiler:

```
./configure CC="cc -Ae -D_XOPEN_SOURCE=500"
```

and if that doesn't work, install pre-built binaries of GCC for HP-UX.

On OSF/1 a.k.a. Tru64, some versions of the default C compiler cannot parse its '<wchar.h>' header file. The option '-nodtk' can be used as a workaround. If GNU CC is not installed, it is therefore recommended to try

```
./configure CC="cc"
```

and if that doesn't work, try

```
./configure CC="cc -nodtk"
```

On Solaris, don't put '/usr/ucb' early in your 'PATH'. This directory contains several dysfunctional programs; working variants of these programs are available in '/usr/bin'. So, if you need '/usr/ucb' in your 'PATH', put it *after* '/usr/bin'.

On Haiku, software installed for all users goes in '/boot/common', not '/usr/local'. It is recommended to use the following options:

```
./cmake -DCMAKE_INSTALL_PREFIX=/boot/common
```

9.10 Specifying the System Type

There may be some features 'configure' cannot figure out automatically, but needs to determine by the type of machine the package will run on. Usually, assuming the package is built to be run on the *same* architectures, 'configure' can figure that out, but if it prints a message saying it cannot guess the machine type, give it the '--build=TYPE' option. TYPE can either be a short name for the system type, such as 'sun4', or a canonical name which has the form CPU-COMPANY-SYSTEM

where SYSTEM can have one of these forms:

- OS

- KERNEL-OS

See the file `'config.sub'` for the possible values of each field. If `'config.sub'` isn't included in this package, then this package doesn't need to know the machine type.

If you are *building* compiler tools for cross-compiling, you should use the option `'--target=TYPE'` to select the type of system they will produce code for.

If you want to use a cross compiler, that generates code for a platform different from the build platform, you should specify the "host" platform (i.e., that on which the generated programs will eventually be run) with `'--host=TYPE'`.

9.11 Sharing Defaults

If you want to set default values for `'configure'` scripts to share, you can create a site shell script called `'config.site'` that gives default values for variables like `'CC'`, `'cache_file'`, and `'prefix'`. `'configure'` looks for `'PREFIX/share/config.site'` if it exists, then `'PREFIX/etc/config.site'` if it exists. Or, you can set the `'CONFIG_SITE'` environment variable to the location of the site script. A warning: not all `'configure'` scripts look for a site script.

9.12 Defining Variables

Variables not defined in a site shell script can be set in the environment passed to `'configure'`. However, some packages may run `configure` again during the build, and the customized values of these variables may be lost. In order to avoid this problem, you should set them in the `'configure'` command line, using `'VAR=value'`. For example:

```
./configure CC=/usr/local2/bin/gcc
```

causes the specified `'gcc'` to be used as the C compiler (unless it is overridden in the site shell script).

Unfortunately, this technique does not work for `'CONFIG_SHELL'` due to an Autoconf bug. Until the bug is fixed you can use this workaround:

```
CONFIG_SHELL=/bin/bash /bin/bash ./configure CONFIG_SHELL=/bin/bash
```

9.13 'cmake' Invocation

`'cmake'` recognizes the following options to control how it operates.

- `'--help'`, `'-h'` print a summary of all of the options to `'configure'`, and exit.
- `'--help=short'`, `'--help=recursive'` print a summary of the options unique to this package's `'configure'`, and exit. The `'short'` variant lists options used only in the top level, while the `'recursive'` variant lists options also present in any nested packages.
- `'--version'`, `'-V'` print the version of Autoconf used to generate the `'configure'` script, and exit.
- `'--cache-file=FILE'` enable the cache: use and save the results of the tests in `FILE`, traditionally `'config.cache'`. `FILE` defaults to `'/dev/null'` to disable caching.

- '--config-cache', '-C' alias for '--cache-file=config.cache'.
- '--quiet', '--silent', '-q' do not print messages saying which checks are being made. To suppress all normal output, redirect it to '/dev/null' (any error messages will still be shown).
- '--srcdir=DIR' look for the package's source code in directory DIR. Usually 'configure' can determine that directory automatically.
- '--prefix=DIR' use DIR as the installation prefix.

See also

[Installation Names](#) for more details, including other options available for fine-tuning the installation locations.

- '--no-create', '-n' run the configure checks, but stop before creating any output files.

'cmake' also accepts some other, not widely useful, options. Run 'cmake --help' for more details.

The 'cmake' script produces an output like this:

```
cmake -DCMAKE_INSTALL_PREFIX=/home/user/dev/deliveries/trademgen-99.99.99 -DLIB_SUFFIX=64 -DCMAKE_BUILD_TYPE:S
-- The C compiler identification is GNU
-- The CXX compiler identification is GNU
-- Check for working C compiler: /usr/lib64/ccache/gcc
-- Check for working C compiler: /usr/lib64/ccache/gcc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: /usr/lib64/ccache/c++
-- Check for working CXX compiler: /usr/lib64/ccache/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Requires Git without specifying any version
-- Current Git revision name: 4856624ea4978b3b2bfe26cb6702cce0be531084 trunk
-- Requires PythonLibs-2.6
-- Found PythonLibs: /usr/lib64/libpython2.7.so (Required is at least version "2.6")
-- Found PythonLibs 2.7
-- Requires Boost-1.41
-- Boost version: 1.46.0
-- Found the following Boost libraries:
--   program_options
--   date_time
--   iostreams
--   serialization
--   filesystem
--   unit_test_framework
--   python
-- Found Boost version: 1.46.0
-- Found BoostWrapper: /usr/include (Required is at least version "1.41")
-- Requires MySQL without specifying any version
-- Using mysql-config: /usr/bin/mysql_config
-- Found MySQL: /usr/lib64/mysql/libmysqlclient.so
-- Found MySQL version: 5.5.14
-- Requires SOCI-3.0
-- Using soci-config: /usr/bin/soci-config
-- SOCI headers are buried
-- Found SOCI: /usr/lib64/libsoci_core.so (Required is at least version "3.0")
-- Found SOCIMySQL: /usr/lib64/libsoci_mysql.so (Required is at least version "3.0")
-- Found SOCI with MySQL back-end support version: 3.0.0
-- Requires StdAir-0.35
-- Found StdAir version: 0.36.2
-- Requires Doxygen without specifying any version
-- Found Doxygen: /usr/bin/doxygen
-- Found DoxygenWrapper: /usr/bin/doxygen
-- Found Doxygen version: 1.7.4
-- Had to set the linker language for 'trademgenlib' to CXX
-- Had to set the linker language for 'pytrademgenlib' to CXX
-- Test 'TrademgenTest' to be built with 'DemandGenerationTestSuite.cpp'
```

```

--
-- =====
-- ---      Project Information      ---
-- -----
-- PROJECT_NAME ..... : trademgen
-- PACKAGE_PRETTY_NAME ..... : TraDemGen
-- PACKAGE ..... : trademgen
-- PACKAGE_NAME ..... : TRADEMGEN
-- PACKAGE_BRIEF ..... : C++ Simulated Travel Demand Generation Library
-- PACKAGE_VERSION ..... : 99.99.99
-- GENERIC_LIB_VERSION ..... : 99.99.99
-- GENERIC_LIB_SOVERSION ..... : 99.99
--
-- -----
-- ---      Build Configuration      ---
-- -----
-- Modules to build ..... : trademgen
-- Libraries to build/install ..... : trademgenlib;pytrademgenlib
-- Binaries to build/install ..... : trademgen;trademgen_with_db;pytrademgen.py
-- Modules to test ..... : trademgen
-- Binaries to test ..... : TrademgenTesttst
--
-- * Module ..... : trademgen
--   + Layers to build ..... : .;basic;bom;factory;command;service
--   + Dependencies on other layers :
--   + Libraries to build/install . : trademgenlib;pytrademgenlib
--   + Executables to build/install : trademgen;trademgen_with_db;pytrademgen.py
--   + Tests to perform ..... : TrademgenTesttst
--
-- BUILD_SHARED_LIBS ..... : ON
-- CMAKE_BUILD_TYPE ..... : Debug
-- * CMAKE_C_FLAGS ..... :
-- * CMAKE_CXX_FLAGS ..... : -Wall -Werror
-- * BUILD_FLAGS ..... :
-- * COMPILE_FLAGS ..... :
-- CMAKE_MODULE_PATH ..... : /home/user/dev/sim/trademgen/trademgengithub/config/
-- CMAKE_INSTALL_PREFIX ..... : /home/user/dev/deliveries/trademgen-99.99.99
--
-- * Doxygen:
--   - DOXYGEN_VERSION ..... : 1.7.4
--   - DOXYGEN_EXECUTABLE ..... : /usr/bin/doxygen
--   - DOXYGEN_DOT_EXECUTABLE ..... : /usr/bin/dot
--   - DOXYGEN_DOT_PATH ..... : /usr/bin
--
-- -----
-- ---      Installation Configuration      ---
-- -----
-- INSTALL_LIB_DIR ..... : /home/user/dev/deliveries/trademgen-99.99.99/lib64
-- INSTALL_BIN_DIR ..... : /home/user/dev/deliveries/trademgen-99.99.99/bin
-- INSTALL_INCLUDE_DIR ..... : /home/user/dev/deliveries/trademgen-99.99.99/include
-- INSTALL_DATA_DIR ..... : /home/user/dev/deliveries/trademgen-99.99.99/share
-- INSTALL_SAMPLE_DIR ..... : /home/user/dev/deliveries/trademgen-99.99.99/share/trademgen/samples
-- INSTALL_DOC ..... : ON
--
-- -----
-- ---      Packaging Configuration      ---
-- -----
-- CPACK_PACKAGE_CONTACT ..... : Denis Arnaud <denis_arnaud - at - users dot sourceforge dot net>
-- CPACK_PACKAGE_VENDOR ..... : Denis Arnaud
-- CPACK_PACKAGE_VERSION ..... : 99.99.99
-- CPACK_PACKAGE_DESCRIPTION_FILE . : /home/user/dev/sim/trademgen/trademgengithub/README
-- CPACK_RESOURCE_FILE_LICENSE .... : /home/user/dev/sim/trademgen/trademgengithub/COPYING
-- CPACK_GENERATOR ..... : TBZ2
-- CPACK_DEBIAN_PACKAGE_DEPENDS ... :
-- CPACK_SOURCE_GENERATOR ..... : TBZ2;TGZ
-- CPACK_SOURCE_PACKAGE_FILE_NAME . : trademgen-99.99.99
--
-- -----
-- ---      External libraries      ---
-- -----
--
-- * Python:

```

```
-- - PYTHONLIBS_VERSION ..... : 2.7
-- - PYTHON_LIBRARIES ..... : /usr/lib64/libpython2.7.so
-- - PYTHON_INCLUDE_PATH ..... : /usr/include/python2.7
-- - PYTHON_INCLUDE_DIRS ..... : /usr/include/python2.7
-- - PYTHON_DEBUG_LIBRARIES ..... :
-- - Python_ADDITIONAL_VERSIONS . :
--
-- * Boost:
-- - Boost_VERSION ..... : 104600
-- - Boost_LIB_VERSION ..... : 1_46
-- - Boost_HUMAN_VERSION ..... : 1.46.0
-- - Boost_INCLUDE_DIRS ..... : /usr/include
-- - Boost required components .. : program_options;date_time;iostreams;serialization;filesystem;unit_test_f
-- - Boost required libraries ... : optimized;/usr/lib64/libboost_iostreams-mt.so;debug;/usr/lib64/libboost_
--
-- * MySQL:
-- - MYSQL_VERSION ..... : 5.5.14
-- - MYSQL_INCLUDE_DIR ..... : /usr/include/mysql
-- - MYSQL_LIBRARIES ..... : /usr/lib64/mysql/libmysqlclient.so
--
-- * SOCI:
-- - SOCI_VERSION ..... : 3.0.0
-- - SOCI_INCLUDE_DIR ..... : /usr/include/soci
-- - SOCI_MYSQL_INCLUDE_DIR ..... : /usr/include/soci
-- - SOCI_LIBRARIES ..... : /usr/lib64/libsoci_core.so
-- - SOCI_MYSQL_LIBRARIES ..... : /usr/lib64/libsoci_mysql.so
--
-- * StdAir:
-- - STDAIR_VERSION ..... : 0.36.2
-- - STDAIR_BINARY_DIRS ..... : /home/user/dev/deliveries/stdair-0.36.2/bin
-- - STDAIR_EXECUTABLES ..... : stdair
-- - STDAIR_LIBRARY_DIRS ..... : /home/user/dev/deliveries/stdair-0.36.2/lib64
-- - STDAIR_LIBRARIES ..... : stdairlib;stdairuiclib
-- - STDAIR_INCLUDE_DIRS ..... : /home/user/dev/deliveries/stdair-0.36.2/include
-- - STDAIR_SAMPLE_DIR ..... : /home/user/dev/deliveries/stdair-0.36.2/share/stdair/samples
--
-- Change a value with: cmake -D<Variable>=<Value>
-- =====
--
-- Configuring done
-- Generating done
-- Build files have been written to: /home/user/dev/sim/trademgen/trademgengithub/build
```

It is recommended that you check if your library has been compiled and linked properly and works as expected. To do so, you should execute the testing process 'make check'. As a result, you should obtain a similar report:

```
[ 0%] Built target hdr_cfg_trademgen
[ 94%] Built target trademgenlib
[100%] Built target TrademgenTesttst
Scanning dependencies of target check_trademgentst
Test project /home/user/dev/sim/trademgen/trademgengithub/build/test/trademgen
  Start 1: TrademgenTesttst
1/1 Test #1: TrademgenTesttst ..... Passed    0.37 sec

100% tests passed, 0 tests failed out of 1

Total Test time (real) = 10.82 sec
[100%] Built target check_trademgentst
Scanning dependencies of target check
[100%] Built target check
```

Check if all the executed tests PASSED. If not, please contact us by filling a [bug-report](#).

Finally, you should install the compiled and linked library, include files and (optionally) HTML and PDF documentation by typing:

```
make install
```

Depending on the PREFIX settings during configuration, you might need the root (administrator) access to perform this step.

Eventually, you might invoke the following command

```
make clean
```

to remove all files created during compilation process, or even

```
cd ~/dev/sim/trademgengit
rm -rf build && mkdir build
cd build
```

to remove everything.

10 Linking with TraDemGen

10.1 Table of Contents

- [Introduction](#)
- [Using the pkg-config command](#)
- [Using the trademgen-config script](#)
- [M4 macro for the GNU Autotools](#)
- [Using TraDemGen with dynamic linking](#)

10.2 Introduction

There are two convenient methods of linking your programs with the TraDemGen library. The first one employs the `'pkg-config'` command (see <http://pkgconfig.freedesktop.org/>), whereas the second one uses `'trademgen-config'` script. These methods are shortly described below.

10.3 Using the pkg-config command

`'pkg-config'` is a helper tool used when compiling applications and libraries. It helps you insert the correct compiler and linker options. The syntax of the `'pkg-config'` is as follows:

```
pkg-config <options> <library_name>
```

For instance, assuming that you need to compile an TraDemGen based program `'my_prog.cpp'`, you should use the following command:

```
g++ `pkg-config --cflags trademgen` -o my_prog my_prog.cpp `pkg-config --libs trademgen`
```

For more information see the `'pkg-config'` man pages.

10.4 Using the trademgen-config script

TraDemGen provides a shell script called `trademgen-config`, which is installed by default in `'$prefix/bin'` (`'/usr/local/bin'`) directory. It can be used to simplify compilation and linking of TraDemGen based programs. The usage of this script is quite similar to the usage of the `'pkg-config'` command.

Assuming that you need to compile the program `'my_prog.cpp'` you can now do that with the following command:

```
g++ `trademgen-config --cflags` -o my_prog_opt my_prog.cpp `trademgen-config --
libs`
```

A list of `'trademgen-config'` options can be obtained by typing:

```
trademgen-config --help
```

If the `'trademgen-config'` command is not found by your shell, you should add its location `'$prefix/bin'` to the `PATH` environment variable, e.g.:

```
export PATH=/usr/local/bin:$PATH
```

10.5 M4 macro for the GNU Autotools

A M4 macro file is delivered with TraDemGen, namely `'trademgen.m4'`, which can be found in, e.g., `'/usr/share/aclocal'`. When used by a `'configure'` script, thanks to the `'AM_PATH_TraDemGen'` macro (specified in the M4 macro file), the following Makefile variables are then defined:

- `'TraDemGen_VERSION'` (e.g., defined to 0.23.0)
- `'TraDemGen_CFLAGS'` (e.g., defined to `'-I${prefix}/include'`)
- `'TraDemGen_LIBS'` (e.g., defined to `'-L${prefix}/lib -ltrademgen'`)

10.6 Using TraDemGen with dynamic linking

When using static linking some of the library routines in TraDemGen are copied into your executable program. This can lead to unnecessary large executables. To avoid having too large executable files you may use dynamic linking instead. Dynamic linking means that the actual linking is performed when the program is executed. This requires that the system is able to locate the shared TraDemGen library file during your program execution. If you install the TraDemGen library using a non-standard prefix, the `'LD_LIBRARY_PATH'` environment variable might be used to inform the linker of the dynamic library location, e.g.:

```
export LD_LIBRARY_PATH=<TraDemGen installation prefix>/lib:$LD_LIBRARY_PATH
```

11 Test Rules

This section describes how the functionality of the TraDemGen library should be verified. In the `'test/trademgen'` subdirectory, test source files are provided. All functionality should be tested using these test source files.

11.1 The Test Source Files

Each new TraDemGen module/class should be accompanied with a test source file. The test source file is an implementation in C++ that tests the functionality of a function/class or a group of functions/classes called test suites. The test source file should test relevant parameter settings and input/output relations to guarantee correct functionality of the corresponding classes/functions. The test source files should be maintained using version control and updated whenever new functionality is added to the TraDemGen library.

The test source file should print relevant data to a standard output that can be used to verify the functionality. All relevant parameter settings should be tested.

The test source file should be placed in the `'test/trademgen'` subdirectory and should have a name ending with `'TestSuite.cpp'`.

11.2 The Reference File

Consider a test source file named `'YieldTestSuite.cpp'`. A reference file named `'YieldTestSuite.ref'` should accompany the test source file. The reference file contains a reference printout of the standard output generated when running the test program. The reference file should be maintained using version control and updated according to the test source file.

11.3 Testing TraDemGen Library

One can compile and execute all test programs from the `'test/trademgen'` sub-directory by typing:

```
% make check
```

after successful compilation of the TraDemGen library.

12 Users Guide

12.1 Table of Contents

- [Introduction](#)
- [Get Started](#)
 - [Get the TraDemGen library](#)
 - [Build the TraDemGen project](#)
 - [Build and Run the Tests](#)
 - [Install the TraDemGen Project \(Binaries, Documentation\)](#)
- [Exploring the Predefined BOM Tree](#)
 - [Demand Stream Engine BOM Tree](#)
- [Extending the BOM Tree](#)

12.2 Introduction

The TraDemGen library contains classes for yield rule management. This document does not cover all the aspects of the TraDemGen library. It does however explain the most important things you need to know in order to start using TraDemGen.

12.3 Get Started

12.3.1 Get the TraDemGen library

12.3.2 Build the TraDemGen project

To run the configuration script the first time, go to the top directory (where the TraDemGen package has been unpacked), and issue either of the following two commands, depending on whether the TraDemGen project has been checked out from the Subversion repository or downloaded as a tar-ball package from the Sourceforge Web site:

- `./autogen.sh`
- `./configure`

12.3.3 Build and Run the Tests

12.3.4 Install the TraDemGen Project (Binaries, Documentation)

12.4 Exploring the Predefined BOM Tree

TraDemGen predefines a BOM (Business Object Model) tree specific to the airline IT arena.

12.4.1 Demand Stream Engine BOM Tree

- [TRADEMGEN::DemandStream](#)

12.5 Extending the BOM Tree

13 Supported Systems

13.1 Table of Contents

- [Introduction](#)
- [TraDemGen 3.10.x](#)
 - [Linux Systems](#)
 - * [Fedora Core 4 with ATLAS](#)
 - * [Gentoo Linux with ACML](#)
 - * [Gentoo Linux with ATLAS](#)
 - * [Gentoo Linux with MKL](#)
 - * [Gentoo Linux with NetLib's BLAS and LAPACK](#)
 - * [Red Hat Enterprise Linux with TraDemGen External](#)
 - * [SUSE Linux 10.0 with NetLib's BLAS and LAPACK](#)
 - * [SUSE Linux 10.0 with MKL](#)
 - [Windows Systems](#)
 - * [Microsoft Windows XP with Cygwin](#)
 - * [Microsoft Windows XP with Cygwin and ATLAS](#)
 - * [Microsoft Windows XP with Cygwin and ACML](#)
 - * [Microsoft Windows XP with MinGW, MSYS and ACML](#)
 - * [Microsoft Windows XP with MinGW, MSYS and TraDemGen External](#)
 - * [Microsoft Windows XP with MS Visual C++ and Intel MKL](#)
 - [Unix Systems](#)
 - * [SunOS 5.9 with TraDemGen External](#)
- [TraDemGen 3.9.1](#)
- [TraDemGen 3.9.0](#)
- [TraDemGen 3.8.1](#)

13.2 Introduction

This page is intended to provide a list of TraDemGen supported systems, i.e. the systems on which configuration, installation and testing process of the TraDemGen library has been successful. Results are grouped based on minor release number. Therefore, only the latest tests for bug-fix releases are included. Besides, the information on this page is divided into sections dependent on the operating system.

Where necessary, some extra information is given for each tested configuration, e.g. external libraries installed, configuration commands used, etc.

If you manage to compile, install and test the TraDemGen library on a system not mentioned below, please let us know, so we could update this database.

13.3 TraDemGen 3.10.x

13.3.1 Linux Systems

13.3.1.1 Fedora Core 4 with ATLAS

- **Platform:** Intel Pentium 4
- **Operating System:** Fedora Core 4 (x86)
- **Compiler:** g++ (GCC) 4.0.2 20051125
- **TraDemGen release:** 3.10.0
- **External Libraries:** From FC4 distribution:
 - fftw3.i386-3.0.1-3
 - fftw3-devel.i386-3.0.1-3
 - atlas-sse2.i386-3.6.0-8.fc4
 - atlas-sse2-devel.i386-3.6.0-8.fc4
 - blas.i386-3.0-35.fc4
 - lapack.i386-3.0-35.fc4
- **Tests Status:** All tests PASSED
- **Comments:** TraDemGen configured with:


```
% CXXFLAGS="-O3 -pipe -march=pentium4" ./configure
```
- **Date:** March 7, 2006
- **Tester:** Tony Ottosson

13.3.1.2 Gentoo Linux with ACML

- **Platform:** AMD Sempron 3000+
- **Operating System:** Gentoo Linux 2006.0 (x86 arch)
- **Compiler(s):** g++ (GCC) 3.4.5
- **TraDemGen release:** 3.10.1
- **External Libraries:** Compiled and installed from portage tree:
 - sci-libs/acml-3.0.0
- **Tests Status:** All tests PASSED
- **Comments:** BLAS and LAPACK libs set by using the following system commands:


```
% eselect blas set ACML
% eselect lapack set ACML
```

TraDemGen configured with:

```
% export CPPFLAGS="-I/usr/include/acml"
% ./configure --with-blas="-lblas"
```
- **Date:** March 31, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.1.3 Gentoo Linux with ATLAS

- **Platform:** Intel Pentium M Centrino
- **Operating System:** Gentoo Linux 2006.0 (x86)
- **Compiler:** g++ (GCC) 3.4.5
- **TraDemGen release:** 3.10.1
- **External Libraries:** Compiled and installed from portage tree:
 - sci-libs/fftw-3.1
 - sci-libs/blas-atlas-3.6.0-r1
 - sci-libs/lapack-atlas-3.6.0
- **Tests Status:** All tests PASSED
- **Comments:** BLAS and LAPACK libs set by using the following system commands:

```
% eselect blas set ATLAS
% eselect lapack set ATLAS
```

TraDemGen configured with:

```
% ./configure --with-blas="-lblas"
```

- **Date:** March 31, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.1.4 Gentoo Linux with MKL

- **Platform:** Intel Pentium M Centrino
- **Operating System:** Gentoo Linux 2006.0 (x86 arch)
- **Compiler:** g++ (GCC) 3.4.5
- **TraDemGen release:** 3.10.0
- **External Libraries:** Intel Math Kernel Library (MKL) 8.0.1 installed manually in the following directory:
/opt/intel/mkl/8.0.1
- **Tests Status:** All tests PASSED
- **Comments:** TraDemGen configured using the following commands:

```
% export LDFLAGS="-L/opt/intel/mkl/8.0.1/lib/32"
% export CPPFLAGS="-I/opt/intel/mkl/8.0.1/include"
% ./configure
```

- **Date:** February 28, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.1.5 Gentoo Linux with NetLib's BLAS and LAPACK

- **Platform:** Intel Pentium M Centrino
- **Operating System:** Gentoo Linux 2006.0 (x86)
- **Compiler:** g++ (GCC) 3.4.5
- **TraDemGen release:** 3.10.1
- **External Libraries:** Compiled and installed from portage tree:
 - sci-libs/fftw-3.1
 - sci-libs/blas-reference-19940131-r2
 - sci-libs/cblas-reference-20030223
 - sci-libs/lapack-reference-3.0-r2
- **Tests Status:** All tests PASSED
- **Comments:** BLAS and LAPACK libs set by using the following system commands:

```
% blas-config reference
% lapack-config reference
```

TraDemGen configured with:

```
% ./configure --with-blas="-lblas"
```

- **Date:** March 31, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.1.6 Red Hat Enterprise Linux with TraDemGen External

- **Platform:** Intel Pentium 4
- **Operating System:** Red Hat Enterprise Linux AS release 4 (Nahant Update 2)
- **Compiler:** g++ (GCC) 3.4.4 20050721 (Red Hat 3.4.4-2)
- **TraDemGen release:** 3.10.0
- **External Libraries:** BLAS, CBLAS, LAPACK and FFTW libraries from TraDemGen External 2.1.1 package
- **Tests Status:** All tests PASSED
- **Date:** March 7, 2006
- **Tester:** Erik G. Larsson

13.3.1.7 SUSE Linux 10.0 with NetLib's BLAS and LAPACK

- **Platform:** Intel Pentium 4 CPU 3.20GHz (64-bit)
- **Operating System:** SUSE Linux 10.0 (x86_64)
- **Compiler(s):** g++ (GCC) 4.0.2
- **TraDemGen release:** 3.10.0
- **External Libraries:** BLAS, LAPACK and FFTW libraries installed from OpenSuse 10.0 RPM repository:
 - blas-3.0-926
 - lapack-3.0-926
 - fftw3-3.0.1-114

- fftw3-threads-3.0.1-114
- fftw3-devel-3.0.1-114

- **Tests Status:** All tests PASSED
- **Comments:** TraDemGen configured with:

```
% export CXXFLAGS="-m64 -march=nocona -O3 -pipe"
% ./configure --with-lapack="/usr/lib64/liblapack.so.3"
```

- **Date:** March 1, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.1.8 SUSE Linux 10.0 with MKL

- **Platform:** Intel Pentium 4 CPU 3.20GHz (64-bit)
- **Operating System:** SUSE Linux 10.0 (x86_64)
- **Compiler(s):** g++ (GCC) 4.0.2
- **TraDemGen release:** 3.10.0
- **External Libraries:** Intel Math Kernel Library (MKL) 8.0.1 installed manually in the following directory:
/opt/intel/mkl/8.0.1
- **Tests Status:** All tests PASSED
- **Comments:** TraDemGen configured with:

```
% export CXXFLAGS="-m64 -march=nocona -O3 -pipe"
% export LDFLAGS="-L/opt/intel/mkl/8.0.1/lib/em64t"
% export CPPFLAGS="-I/opt/intel/mkl/8.0.1/include"
% ./configure
```

- **Date:** March 1, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2 Windows Systems

13.3.2.1 Microsoft Windows XP with Cygwin

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2, Cygwin 1.5.19-4
- **Compiler(s):** g++ (GCC) 3.4.4 (cygming special)
- **TraDemGen release:** 3.10.1
- **External Libraries:** Installed from Cygwin's repository:
 - fftw-3.0.1-2
 - fftw-dev-3.0.1-1
 - lapack-3.0-4
- **Tests Status:** All tests PASSED
- **Comments:** Only static library can be built. TraDemGen configured with:

```
% ./configure
```

- **Date:** March 31, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2.2 Microsoft Windows XP with Cygwin and ATLAS

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2, Cygwin 1.5.19-4
- **Compiler(s):** g++ (GCC) 3.4.4 (cygming special)
- **TraDemGen release:** 3.10.1
- **External Libraries:** Installed from Cygwin's repository:
 - fftw-3.0.1-2
 - fftw-dev-3.0.1-1

ATLAS BLAS and LAPACK libraries from TraDemGen External 2.1.1 package configured using:

```
% ./configure --enable-atlas --disable-fftw
```

- **Tests Status:** All tests PASSED
- **Comments:** Only static library can be built. TraDemGen configured with:

```
% export LDFLAGS="-L/usr/local/lib"
% ./configure
```

- **Date:** March 31, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2.3 Microsoft Windows XP with Cygwin and ACML

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2, Cygwin 1.5.19-4
- **Compiler(s):** g++ (GCC) 3.4.4 (cygming special)
- **TraDemGen release:** 3.10.2
- **External Libraries:** ACML version 3.1.0 (acml3.1.0-32-win32-g77.exe) installed into a default directory, i.e. "c:\Program Files\AMD\acml3.1.0"
- **Tests Status:** All tests PASSED
- **Comments:** Only static library can be built. TraDemGen configured with:

```
% export LDFLAGS="-L/cygdrive/c/Progra~1/AMD/acml3.1.0/gnu32/lib"
% export CPPFLAGS="-I/cygdrive/c/Progra~1/AMD/acml3.1.0/gnu32/include"
% ./configure --enable-debug
```

- **Date:** May 15, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2.4 Microsoft Windows XP with MinGW, MSYS and ACML

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2, MinGW 5.0.2, MSYS 1.0.10
- **Compiler(s):** g++ (GCC) 3.4.4 (mingw special)
- **TraDemGen release:** 3.10.2
- **External Libraries:** ACML version 3.1.0 (acml3.1.0-32-win32-g77.exe) installed into a default directory, i.e. "c:\Program Files\AMD\acml3.1.0"
- **Tests Status:** All tests PASSED
- **Comments:** Only static library can be built. TraDemGen configured with:


```
% export LDFLAGS="-L/c/Progra~1/AMD/acml3.1.0/gnu32/lib"
% export CPPFLAGS="-I/c/Progra~1/AMD/acml3.1.0/gnu32/include"
% ./configure --enable-debug
```
- **Date:** May 15, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2.5 Microsoft Windows XP with MinGW, MSYS and TraDemGen External

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2, MinGW 5.0.2, MSYS 1.0.10
- **Compiler(s):** g++ (GCC) 3.4.4 (mingw special)
- **TraDemGen release:** 3.10.5
- **External Libraries:** BLAS, CBLAS, LAPACK and FFTW libraries from TraDemGen External 2.2.0 package
- **Tests Status:** All tests PASSED
- **Comments:** Only static library can be built. TraDemGen configured with:


```
% export LDFLAGS="-L/usr/local/lib"
% export CPPFLAGS="-I/usr/local/include"
% export CXXFLAGS="-Wall -O3 -march=athlon-tbird -pipe"
% ./configure --disable-html-doc
```
- **Date:** August 11, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.2.6 Microsoft Windows XP with MS Visual C++ and Intel MKL

- **Platform:** AMD Sempron 3000+
- **Operating System:** Microsoft Windows XP SP2
- **Compiler(s):** Microsoft Visual C++ 2005 .NET
- **TraDemGen release:** 3.10.5
- **External Libraries:** Intel Math Kernel Library (MKL) 8.1 installed manually in the following directory: "C:\Program Files\Intel\MKL\8.1"
- **Tests Status:** Not fully tested. Some TraDemGen based programs compiled and run with success.
- **Comments:** Only static library can be built. TraDemGen built by opening the "win32\trademgen.-vcproj" project file in MSVC++ and executing "Build -> Build Solution" command from menu.
- **Date:** August 11, 2006
- **Tester:** Adam Piatyszek (ediap)

13.3.3 Unix Systems

13.3.3.1 SunOS 5.9 with TraDemGen External

- **Platform:** SUNW, Sun-Blade-100 (SPARC)
- **Operating System:** SunOS 5.9 Generic_112233-10
- **Compiler(s):** g++ (GCC) 3.4.5
- **TraDemGen release:** 3.10.2
- **External Libraries:** BLAS, CBLAS, LAPACK and FFTW libraries from TraDemGen External 2.1.1 package. The following configuration command has been used:

```
% export CFLAGS="-mcpu=ultrasparc -O2 -pipe -funroll-all-loops"
% ./configure
```

- **Tests Status:** All tests PASSED
- **Comments:** TraDemGen configured with:

```
% export LDFLAGS="-L/usr/local/lib"
% export CPPFLAGS="-I/usr/local/include"
% export CXXFLAGS="-mcpu=ultrasparc -O2 -pipe"
% ./configure --enable-debug
```

- **Date:** May 15, 2006
- **Tester:** Adam Piatyszek (ediap)

14 TraDemGen Supported Systems (Previous Releases)

14.1 TraDemGen 3.9.1

14.2 TraDemGen 3.9.0

14.3 TraDemGen 3.8.1

15 Tutorials

15.1 Table of Contents

- [Introduction](#)
 - [Preparing the StdAir Project for Development](#)
- [Build a Predefined BOM Tree](#)
 - [Instantiate the BOM Root Object](#)
 - [Instantiate the \(Airline\) Inventory Object](#)
 - [Link the Inventory Object with the BOM Root](#)
 - [Build Another Airline Inventory](#)
 - [Dump The BOM Tree Content](#)
 - [Result of the Tutorial Program](#)
- [Extend the Pre-Defined BOM Tree](#)
 - [Extend an Airline Inventory Object](#)
 - [Build the Specific BOM Objects](#)
 - [Result of the Tutorial Program](#)

15.2 Introduction

This page contains some tutorial examples that will help you getting started using StdAir. Most examples show how to construct some simple business objects, i.e., instances of the so-named Business Object Model (BOM).

15.2.1 Preparing the StdAir Project for Development

The source code for these examples can be found in the `batches` and `test/stdair` directories. They are compiled along with the rest of the `StdAir` project. See the User Guide ([Users Guide](#)) for more details on how to build the `StdAir` project.

15.3 Build a Predefined BOM Tree

A few steps:

- [Instantiate the BOM Root Object](#)
- [Instantiate the \(Airline\) Inventory Object](#)
- [Link the Inventory Object with the BOM Root](#)

15.3.1 Instantiate the BOM Root Object

First, a BOM root object (i.e., a root for all the classes in the project) is instantiated by the `stdair::STD-AIR_ServiceContext` context object, when the `stdair::STDAIR_Service` is itself instantiated. The corresponding `StdAir` type (class) is `stdair::BomRoot`.

In the following sample, that object is named `ioBomRoot`, and is given as input/output parameter of the `stdair::CmdBomManager::buildSampleBom()` method:

15.3.2 Instantiate the (Airline) Inventory Object

An airline inventory object can then be instantiated. Let us give it the "BA" airline code (corresponding to [British Airways](#)) as the object key. That is, an object (let us name it `lBAKey`) of type (class) `stdair::Inventory-Key` has first to be instantiated.

Thanks to that key, an airline inventory object, i.e. of type (class) `stdair::Inventory`, can be instantiated. Let us name that airline inventory object `lBAInv`.

15.3.3 Link the Inventory Object with the BOM Root

Then, both objects have to be linked: the airline inventory object (`stdair::Inventory`) has to be linked with the root of the BOM tree (`stdair::BomRoot`). That operation is as simple as using the `stdair::FacBom-Manager::addToListAndMap()` method:

15.3.4 Build Another Airline Inventory

Another airline inventory object, corresponding to the Air France (**Air France**) company, is instantiated the same way:

See the corresponding full program (`cmd_bom_manager_cpp`) for more details.

15.3.5 Dump The BOM Tree Content

From the `BomRoot` (of type `stdair::BomRoot`) object instance, the list of airline inventories (of type `stdair::Inventory`) can then be retrieved...

... and browsed:

See the corresponding full program (`bom_display_cpp`) for more details.

15.3.6 Result of the Tutorial Program

When the `stdair.cpp` program is run (with the `-b` option), the output should look like:

See the corresponding full program (`batch_stdair_cpp`) for more details.

15.4 Extend the Pre-Defined BOM Tree

Now that we master how to instantiate the pre-defined `StdAir` classes, let us see how to extend that BOM.

15.4.1 Extend an Airline Inventory Object

For instance, let us assume that some (IT) provider (e.g., you) would like to have a specific implementation of the `Inventory` object. The corresponding class has just to extend the `stdair::Inventory` class:

The STL containers have to be defined accordingly too:

See the full class definition (`test_archi_inv_hpp`) and implementation (`test_archi_inv_cpp`) for more details.

15.4.2 Build the Specific BOM Objects

The BOM root object (`stdair::BomRoot`) is instantiated the classical way:

Then, the specific implementation of the airline inventory object (`myprovider::Inventory`) can be instantiated the same way as a standard `Inventory` (`stdair::Inventory`) would be:

Then, the specific implementation of the airline inventory object (`myprovider::Inventory`) is linked to the root of the BOM tree (`stdair::BomRoot`) the same way as the standard `Inventory` (`stdair::Inventory`) would be:

Another specific airline inventory object is instantiated the same way:

From the `BomRoot` (of type `stdair::BomRoot`) object instance, the list of specific airline inventories (of type `stdair::Inventory`) can then be retrieved...

... and browsed:

15.4.3 Result of the Tutorial Program

When this program is run, the output should look like:

See the corresponding full program (`StandardAirlineITTestSuite.cpp`) for more details.

16 Command-Line Test to Demonstrate How To Use TraDemGen elements

```

*/
// //////////////////////////////////////
// Import section
// //////////////////////////////////////
// STL
#include <sstream>
#include <fstream>
#include <map>
#include <cmath>
// Boost Unit Test Framework (UTF)
#define BOOST_TEST_DYN_LINK
#define BOOST_TEST_MAIN
#define BOOST_TEST_MODULE DemandGenerationTest
#include <boost/test/unit_test.hpp>
// StdAir
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/basic/BasLogParams.hpp>
#include <stdair/basic/BasDBParams.hpp>
#include <stdair/basic/BasFileMgr.hpp>
#include <stdair/basic/ProgressStatusSet.hpp>
#include <stdair/bom/EventStruct.hpp>
#include <stdair/bom/EventQueue.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/service/Logger.hpp>
// TraDemGen
#include <trademgen/TRADEMGEN_Service.hpp>
#include <trademgen/bom/DemandStreamKey.hpp>
#include <trademgen/config/trademgen-paths.hpp>

namespace boost_utf = boost::unit_test;

```

```

// (Boost) Unit Test XML Report
std::ofstream utfReportStream ("DemandGenerationTestSuite_utfresults.xml");

struct UnitTestConfig {
    UnitTestConfig() {
        boost_utf::unit_test_log.set_stream (utfReportStream);
        boost_utf::unit_test_log.set_format (boost_utf::XML);
        boost_utf::unit_test_log.set_threshold_level (boost_utf::log_test_units);
        //boost_utf::unit_test_log.set_threshold_level
        (boost_utf::log_successful_tests);
    }

    ~UnitTestConfig() {
    }
};

// Specific type definitions
typedef std::pair<stdair::Count_T, stdair::Count_T> NbOfEventsPair_T;
typedef std::map<const stdair::DemandStreamKeyStr_T,
                NbOfEventsPair_T> NbOfEventsByDemandStreamMap_T;

// //////////// Main: Unit Test Suite ////////////

// Set the UTF configuration (re-direct the output to a specific file)
BOOST_GLOBAL_FXTURE (UnitTestConfig);

// Start the test suite
BOOST_AUTO_TEST_SUITE (master_test_suite)

BOOST_AUTO_TEST_CASE (trademgen_simple_simulation_test) {

    // Seed for the random generation
    const stdair::RandomSeed_T lRandomSeed = stdair::DEFAULT_RANDOM_SEED;

// Input file name
    const stdair::Filename_T lInputFilename (STDAIR_SAMPLE_DIR "/demand01.csv");

    // Check that the file path given as input corresponds to an actual file
    const bool doesExistAndIsReadable =
        stdair::BasFileMgr::doesExistAndIsReadable (lInputFilename);
    BOOST_CHECK_MESSAGE (doesExistAndIsReadable == true,
        "The '" << lInputFilename
        << "' input file can not be open and read");

    // Output log File
    const stdair::Filename_T lLogFilename ("DemandGenerationTestSuite.log");

    // Set the log parameters
    std::ofstream logOutputFile;
    // open and clean the log outputfile
    logOutputFile.open (lLogFilename.c_str());
    logOutputFile.clear();

    // Initialise the TraDemGen service object
    const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
    TRADEMGEN::TRADEMGEN_Service trademgenService (lLogParams, lRandomSeed);

    // Create the DemandStream objects, and insert them within the BOM tree
    BOOST_CHECK_NO_THROW (trademgenService.parseAndLoad (lInputFilename));

    NbOfEventsByDemandStreamMap_T lNbOfEventsMap;
    lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
        value_type ("SIN-HND 2010-Feb-08 Y",
            NbOfEventsPair_T (1, 10)));
    lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
        value_type ("SIN-HND 2010-Feb-09 Y",
            NbOfEventsPair_T (1, 10)));
    lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
        value_type ("SIN-BKK 2010-Feb-08 Y",
            NbOfEventsPair_T (1, 10)));
    lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
        value_type ("SIN-BKK 2010-Feb-09 Y",
            NbOfEventsPair_T (1, 10)));

    // Total number of events, for all the demand streams: 3
    stdair::Count_T lRefExpectedNbOfEvents (40);

    // Retrieve the expected (mean value of the) number of events to be
    // generated
    const stdair::Count_T& lExpectedNbOfEventsToBeGenerated =
        trademgenService.getExpectedTotalNumberOfRequestsToBeGenerated();

    BOOST_CHECK_EQUAL (lRefExpectedNbOfEvents,
        std::floor (lExpectedNbOfEventsToBeGenerated));

    BOOST_CHECK_MESSAGE (lRefExpectedNbOfEvents ==

```

```

        std::floor (lExpectedNbOfEventsToBeGenerated),
        "Expected total number of requests to be generated: "
        << lExpectedNbOfEventsToBeGenerated
        << " (=> "
        << std::floor (lExpectedNbOfEventsToBeGenerated)
        << "). Reference value: " << lRefExpectedNbOfEvents);

// Generate the date time of the requests with the statistic order method.
stdair::DemandGenerationMethod lDemandGenerationMethod (
    stdair::DemandGenerationMethod::STA_ORD);

const stdair::Count_T& lActualNbOfEventsToBeGenerated =
    trademgenService.generateFirstRequests(lDemandGenerationMethod);

// DEBUG
STDAIR_LOG_DEBUG ("Expected number of events: "
    << lExpectedNbOfEventsToBeGenerated << ", actual: "
    << lActualNbOfEventsToBeGenerated);

// Total number of events, for all the demand streams: 40
const stdair::Count_T lRefActualNbOfEvents (40);
BOOST_CHECK_EQUAL (lRefActualNbOfEvents, lActualNbOfEventsToBeGenerated);

BOOST_CHECK_MESSAGE (lRefActualNbOfEvents == lActualNbOfEventsToBeGenerated,
    "Actual total number of requests to be generated: "
    << lExpectedNbOfEventsToBeGenerated
    << " (=> "
    << std::floor (lExpectedNbOfEventsToBeGenerated)
    << "). Reference value: " << lRefActualNbOfEvents);

const bool isQueueDone = trademgenService.isQueueDone();
BOOST_REQUIRE_MESSAGE (isQueueDone == false,
    "The event queue should not be empty. You may check "
    << "the input file: '" << lInputFilename << "'");

stdair::Count_T idx = 1;
while (trademgenService.isQueueDone() == false) {

    // Get the next event from the event queue
    stdair::EventStruct lEventStruct;
    stdair::ProgressStatusSet lPPS = trademgenService.popEvent (lEventStruct);

    // DEBUG
    STDAIR_LOG_DEBUG ("Popped event: '" << lEventStruct.describe() << "'");

    // Extract the corresponding demand/booking request
    const stdair::BookingRequestStruct& lPoppedRequest =
        lEventStruct.getBookingRequest();

    // DEBUG
    STDAIR_LOG_DEBUG ("Popped booking request: '"
        << lPoppedRequest.describe() << "'");

    // Retrieve the corresponding demand stream
    const stdair::DemandGeneratorKey_T& lDemandStreamKey =
        lPoppedRequest.getDemandGeneratorKey();

    // Check that the number of booking requests to be generated are correct
    const NbOfEventsByDemandStreamMap_T::iterator itNbOfEventsMap =
        lNbOfEventsMap.find (lDemandStreamKey);
    BOOST_REQUIRE_MESSAGE (itNbOfEventsMap != lNbOfEventsMap.end(),
        "The demand stream key '" << lDemandStreamKey
        << "' is not expected in that test");

    const NbOfEventsPair_T& lNbOfEventsPair = itNbOfEventsMap->second;
    stdair::Count_T lCurrentNbOfEvents = lNbOfEventsPair.first;
    const stdair::Count_T& lExpectedTotalNbOfEvents = lNbOfEventsPair.second;

    // Assess whether more events should be generated for that demand stream
    const bool stillHavingRequestsToBeGenerated = trademgenService.
        stillHavingRequestsToBeGenerated (lDemandStreamKey, lPPS,
            lDemandGenerationMethod);

    if (lCurrentNbOfEvents == 1) {
        const stdair::ProgressStatus& lDemandStreamProgressStatus =
            lPPS.getSpecificGeneratorStatus();
        const stdair::Count_T& lNbOfRequests =
            lDemandStreamProgressStatus.getExpectedNb();

        BOOST_CHECK_EQUAL (lNbOfRequests, lExpectedTotalNbOfEvents);
        BOOST_CHECK_MESSAGE (lNbOfRequests == lExpectedTotalNbOfEvents,
            "[" << lDemandStreamKey
            << "] Total number of requests to be generated: "
            << lNbOfRequests << "). Expected value: "
            << lExpectedTotalNbOfEvents);
    }
}

```

```

// DEBUG
STDAIR_LOG_DEBUG ("=> [" << lDemandStreamKey << "]"[" << lCurrentNbOfEvents
<< "/" << lExpectedTotalNbOfEvents
<< "] is now processed. "
<< "Still generate events for that demand stream? "
<< stillHavingRequestsToBeGenerated);

// If there are still events to be generated for that demand stream,
// generate and add them to the event queue
if (stillHavingRequestsToBeGenerated == true) {
    const stdair::BookingRequestPtr_T lNextRequest_ptr =
        trademgenService.generateNextRequest (lDemandStreamKey,
                                                lDemandGenerationMethod);
    assert (lNextRequest_ptr != NULL);

    const stdair::Duration_T lDuration =
        lNextRequest_ptr->getRequestDateTime()
        - lPoppedRequest.getRequestDateTime();
    BOOST_REQUIRE_GT (lDuration.total_milliseconds(), 0);
    BOOST_REQUIRE_MESSAGE (lDuration.total_milliseconds() > 0,
        "[" << lDemandStreamKey
        << "] The date-time of the generated event ("
        << lNextRequest_ptr->getRequestDateTime()
        << ") is lower than the date-time "
        << "of the current event ("
        << lPoppedRequest.getRequestDateTime() << ")");

    // DEBUG
    STDAIR_LOG_DEBUG ("[" << lDemandStreamKey << "]"[" << lCurrentNbOfEvents
    << "/" << lExpectedTotalNbOfEvents
    << "] Added request: '" << lNextRequest_ptr->describe()
    << "'. Is queue done? "
    << trademgenService.isQueueDone());

    // Keep, within the dedicated map, the current counters of events
    updated.
    ++lCurrentNbOfEvents;
    itNbOfEventsMap->second = NbOfEventsPair_T (lCurrentNbOfEvents,
                                                lExpectedTotalNbOfEvents);
}

// Iterate
++idx;
}
// Compensate for the last iteration
--idx;
//
BOOST_CHECK_EQUAL (idx, lRefActualNbOfEvents);
BOOST_CHECK_MESSAGE (idx == lRefActualNbOfEvents,
    "The total actual number of events is "
    << lRefActualNbOfEvents << ", but " << idx
    << " events have been generated");

trademgenService.reset();

// DEBUG
STDAIR_LOG_DEBUG ("End of the simulation");

// Close the log file
logOutputFile.close();
}

// End the test suite
BOOST_AUTO_TEST_SUITE_END()

/*!

```

17 Directory Hierarchy

17.1 Directories

This directory hierarchy is sorted roughly, but not completely, alphabetically:

test	51
trademgen	52
trademgen	51

basic	49
batches	50
bom	50
command	50
config	50
factory	50
python	50
service	51
ui	52
qt	51
trademgen	51

18 Namespace Index

18.1 Namespace List

Here is a list of all namespaces with brief descriptions:

stdair	
Forward declarations	52
TRADEMGEN	52
TRADEMGEN::DemandParserHelper	60

19 Class Index

19.1 Class Hierarchy

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

```
std::basic_fstream< char >
std::basic_fstream< wchar_t >
std::basic_ifstream< char >
std::basic_ifstream< wchar_t >
std::basic_ios< char >
std::basic_ios< wchar_t >
std::basic_iostream< char >
std::basic_iostream< wchar_t >
std::basic_istream< char >
std::basic_istream< wchar_t >
std::basic_istreamstream< char >
std::basic_istreamstream< wchar_t >
std::basic_ofstream< char >
std::basic_ofstream< wchar_t >
std::basic_ostream< char >
std::basic_ostream< wchar_t >
```


std::basic_ostringstream< char >	
std::basic_ostringstream< wchar_t >	
std::basic_string< char >	
std::basic_string< wchar_t >	
std::basic_stringstream< char >	
std::basic_stringstream< wchar_t >	
BomAbstract	63
TRADEMGEN::DemandStream	93
TRADEMGEN::BomDisplay	63
stdair::CategoricalAttribute< T >	64
TRADEMGEN::CategoricalAttributeLite< T >	66
CmdAbstract	68
TRADEMGEN::DemandFileParser	88
TRADEMGEN::DemandManager	90
TRADEMGEN::DemandParser	92
TRADEMGEN::ContinuousAttribute< T >	68
TRADEMGEN::ContinuousAttributeLite< T >	70
TRADEMGEN::DBManager	72
TRADEMGEN::DefaultMap	76
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >	77
TRADEMGEN::DictionaryManager	112
FacServiceAbstract	115
TRADEMGEN::FacTRADEMGENServiceContext	115
FileNotFoundException	117
TRADEMGEN::DemandInputFileNotFoundException	89
TRADEMGEN::FlagSaver	117
grammar	118
TRADEMGEN::DemandParserHelper::DemandParser	91
KeyAbstract	119
TRADEMGEN::DemandStreamKey	105
TRADEMGEN::DemandParserHelper::ParserSemanticAction	119
TRADEMGEN::DemandParserHelper::doEndDemand	113
TRADEMGEN::DemandParserHelper::storeChannelCode	123
TRADEMGEN::DemandParserHelper::storeChannelProbMass	124

TRADEMGEN::DemandParserHelper::storeDemandMean	126
TRADEMGEN::DemandParserHelper::storeDemandStdDev	127
TRADEMGEN::DemandParserHelper::storeDestination	129
TRADEMGEN::DemandParserHelper::storeDow	130
TRADEMGEN::DemandParserHelper::storeDTD	131
TRADEMGEN::DemandParserHelper::storeDTDProbMass	133
TRADEMGEN::DemandParserHelper::storeFFCode	134
TRADEMGEN::DemandParserHelper::storeFFProbMass	136
TRADEMGEN::DemandParserHelper::storeOrigin	137
TRADEMGEN::DemandParserHelper::storePosCode	138
TRADEMGEN::DemandParserHelper::storePosProbMass	140
TRADEMGEN::DemandParserHelper::storePrefCabin	141
TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd	142
TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart	144
TRADEMGEN::DemandParserHelper::storePrefDepTime	145
TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass	147
TRADEMGEN::DemandParserHelper::storeStayCode	148
TRADEMGEN::DemandParserHelper::storeStayProbMass	149
TRADEMGEN::DemandParserHelper::storeTimeValue	151
TRADEMGEN::DemandParserHelper::storeTimeValueProbMass	152
TRADEMGEN::DemandParserHelper::storeTripCode	153
TRADEMGEN::DemandParserHelper::storeTripProbMass	155
TRADEMGEN::DemandParserHelper::storeWTP	156
RootException	122
TRADEMGEN::TrademgenGenerationException	168
TRADEMGEN::IndexOutOfRangeException	118
ServiceAbstract	123
TRADEMGEN::TRADEMGEN_ServiceContext	166
StructAbstract	158
TRADEMGEN::DemandCharacteristics	82
TRADEMGEN::DemandDistribution	86
TRADEMGEN::DemandStruct	107

TRADEMGEN::RandomGenerationContext	120
TestFixture	158
DemandGenerationTestSuite	89
TRADEMGEN::TRADEMGEN_Abstract	158
TRADEMGEN::DBParams	73
TRADEMGEN::TRADEMGEN_Service	160
TRADEMGEN::Trademgener	167

20 Class Index

20.1 Class List

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

BomAbstract	63
TRADEMGEN::BomDisplay Utility class to display TraDemGen objects with a pretty format	63
stdair::CategoricalAttribute< T > Class modeling the distribution of values that can be taken by a categorical attribute	64
TRADEMGEN::CategoricalAttributeLite< T > Class modeling the distribution of values that can be taken by a categorical attribute	66
CmdAbstract	68
TRADEMGEN::ContinuousAttribute< T >	68
TRADEMGEN::ContinuousAttributeLite< T > Class modeling the distribution of values that can be taken by a continuous attribute	70
TRADEMGEN::DBManager	72
TRADEMGEN::DBParams	73
TRADEMGEN::DefaultMap	76
TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >	77
TRADEMGEN::DemandCharacteristics Class modeling the characteristics of a demand type	82
TRADEMGEN::DemandDistribution Class modeling the distribution of a demand type	86
TRADEMGEN::DemandFileParser	88
DemandGenerationTestSuite	89
TRADEMGEN::DemandInputFileNotFoundException	89
TRADEMGEN::DemandManager Utility class for Demand and DemandStream objects	90

TRADEMGEN::DemandParserHelper::DemandParser	91
TRADEMGEN::DemandParser	
Class wrapping the parser entry point	92
TRADEMGEN::DemandStream	
Class modeling a demand stream	93
TRADEMGEN::DemandStreamKey	105
TRADEMGEN::DemandStruct	107
TRADEMGEN::DictionaryManager	
Class wrapper of dictionary business methods	112
TRADEMGEN::DemandParserHelper::doEndDemand	113
FacServiceAbstract	115
TRADEMGEN::FacTRADEMGENServiceContext	
Factory for creating the TraDemGen service context instance	115
FileNotFoundException	117
TRADEMGEN::FlagSaver	117
grammar	118
TRADEMGEN::IndexOutOfRangeException	118
KeyAbstract	119
TRADEMGEN::DemandParserHelper::ParserSemanticAction	119
TRADEMGEN::RandomGenerationContext	120
RootException	122
ServiceAbstract	123
TRADEMGEN::DemandParserHelper::storeChannelCode	123
TRADEMGEN::DemandParserHelper::storeChannelProbMass	124
TRADEMGEN::DemandParserHelper::storeDemandMean	126
TRADEMGEN::DemandParserHelper::storeDemandStdDev	127
TRADEMGEN::DemandParserHelper::storeDestination	129
TRADEMGEN::DemandParserHelper::storeDow	130
TRADEMGEN::DemandParserHelper::storeDTD	131
TRADEMGEN::DemandParserHelper::storeDTDProbMass	133
TRADEMGEN::DemandParserHelper::storeFFCode	134
TRADEMGEN::DemandParserHelper::storeFFProbMass	136
TRADEMGEN::DemandParserHelper::storeOrigin	137
TRADEMGEN::DemandParserHelper::storePosCode	138

TRADEMGEN::DemandParserHelper::storePosProbMass	140
TRADEMGEN::DemandParserHelper::storePrefCabin	141
TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd	142
TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart	144
TRADEMGEN::DemandParserHelper::storePrefDepTime	145
TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass	147
TRADEMGEN::DemandParserHelper::storeStayCode	148
TRADEMGEN::DemandParserHelper::storeStayProbMass	149
TRADEMGEN::DemandParserHelper::storeTimeValue	151
TRADEMGEN::DemandParserHelper::storeTimeValueProbMass	152
TRADEMGEN::DemandParserHelper::storeTripCode	153
TRADEMGEN::DemandParserHelper::storeTripProbMass	155
TRADEMGEN::DemandParserHelper::storeWTP	156
StructAbstract	158
TestFixture	158
TRADEMGEN::TRADEMGEN_Abstract	158
TRADEMGEN::TRADEMGEN_Service	
Class holding the services related to Travel Demand Generation	160
TRADEMGEN::TRADEMGEN_ServiceContext	
Class holding the context of the Trademgen services	166
TRADEMGEN::Trademgener	167
TRADEMGEN::TrademgenGenerationException	168

21 File Index

21.1 File List

Here is a list of all files with brief descriptions:

test/trademgen/DemandGenerationTestSuite.cpp	170
test/trademgen/DemandGenerationTestSuite.hpp	173
test/trademgen/generateEvents.cpp	174
trademgen/DBParams.hpp	268
trademgen/TRADEMGEN_Abstract.hpp	285
trademgen/TRADEMGEN_Exceptions.hpp	287

trademgen/TRADEMGEN_Service.hpp	287
trademgen/TRADEMGEN_Types.hpp	289
trademgen/basic/BasConst.cpp	175
trademgen/basic/BasConst_DemandGeneration.hpp	176
trademgen/basic/BasConst_TRADEMGEN_Service.hpp	177
trademgen/basic/BasParserTypes.hpp	179
trademgen/basic/CategoricalAttribute.hpp	180
trademgen/basic/CategoricalAttributeLite.hpp	182
trademgen/basic/ContinuousAttribute.hpp	184
trademgen/basic/ContinuousAttributeLite.hpp	186
trademgen/basic/DemandCharacteristics.cpp	189
trademgen/basic/DemandCharacteristics.hpp	191
trademgen/basic/DemandCharacteristicsTypes.hpp	193
trademgen/basic/DemandDistribution.cpp	194
trademgen/basic/DemandDistribution.hpp	195
trademgen/basic/DictionaryManager.cpp	196
trademgen/basic/DictionaryManager.hpp	197
trademgen/basic/RandomGenerationContext.cpp	197
trademgen/basic/RandomGenerationContext.hpp	198
trademgen/batches/trademgen.cpp	202
trademgen/batches/trademgen_with_db.cpp	211
trademgen/bom/BomDisplay.cpp	215
trademgen/bom/BomDisplay.hpp	217
trademgen/bom/DemandStream.cpp	217
trademgen/bom/DemandStream.hpp	225
trademgen/bom/DemandStreamKey.cpp	228
trademgen/bom/DemandStreamKey.hpp	229
trademgen/bom/DemandStreamTypes.hpp	230
trademgen/bom/DemandStruct.cpp	231
trademgen/bom/DemandStruct.hpp	233
trademgen/command/DBManager.cpp	235
trademgen/command/DBManager.hpp	237

trademgen/command/ DemandManager.cpp	238
trademgen/command/ DemandManager.hpp	248
trademgen/command/ DemandParser.cpp	250
trademgen/command/ DemandParser.hpp	251
trademgen/command/ DemandParserHelper.cpp	252
trademgen/command/ DemandParserHelper.hpp	263
trademgen/config/ trademgen-paths.hpp	268
trademgen/factory/ FacTRADEMGENSEerviceContext.cpp	270
trademgen/factory/ FacTRADEMGENSEerviceContext.hpp	271
trademgen/python/ pytrademgen.cpp	272
trademgen/service/ TRADEMGEN_Service.cpp	275
trademgen/service/ TRADEMGEN_ServiceContext.cpp	282
trademgen/service/ TRADEMGEN_ServiceContext.hpp	283
trademgen/ui/qt/trademgen/ main.cpp	290
trademgen/ui/qt/trademgen/ trademgen.cpp	208

22 Directory Documentation

22.1 trademgen/basic/ Directory Reference

Files

- file [BasConst.cpp](#)
- file [BasConst_DemandGeneration.hpp](#)
- file [BasConst_TRADEMGEN_Service.hpp](#)
- file [BasParserTypes.hpp](#)
- file [CategoricalAttribute.hpp](#)
- file [CategoricalAttributeLite.hpp](#)
- file [ContinuousAttribute.hpp](#)
- file [ContinuousAttributeLite.hpp](#)
- file [DemandCharacteristics.cpp](#)
- file [DemandCharacteristics.hpp](#)
- file [DemandCharacteristicsTypes.hpp](#)
- file [DemandDistribution.cpp](#)
- file [DemandDistribution.hpp](#)
- file [DictionaryManager.cpp](#)
- file [DictionaryManager.hpp](#)
- file [RandomGenerationContext.cpp](#)
- file [RandomGenerationContext.hpp](#)

22.2 trademgen/batches/ Directory Reference

Files

- file [trademgen.cpp](#)
- file [trademgen_with_db.cpp](#)

22.3 trademgen/bom/ Directory Reference

Files

- file [BomDisplay.cpp](#)
- file [BomDisplay.hpp](#)
- file [DemandStream.cpp](#)
- file [DemandStream.hpp](#)
- file [DemandStreamKey.cpp](#)
- file [DemandStreamKey.hpp](#)
- file [DemandStreamTypes.hpp](#)
- file [DemandStruct.cpp](#)
- file [DemandStruct.hpp](#)

22.4 trademgen/command/ Directory Reference

Files

- file [DBManager.cpp](#)
- file [DBManager.hpp](#)
- file [DemandManager.cpp](#)
- file [DemandManager.hpp](#)
- file [DemandParser.cpp](#)
- file [DemandParser.hpp](#)
- file [DemandParserHelper.cpp](#)
- file [DemandParserHelper.hpp](#)

22.5 trademgen/config/ Directory Reference

Files

- file [trademgen-paths.hpp](#)

22.6 trademgen/factory/ Directory Reference

Files

- file [FacTRADEMGENServiceContext.cpp](#)
- file [FacTRADEMGENServiceContext.hpp](#)

22.7 trademgen/python/ Directory Reference

Files

- file [pytrademgen.cpp](#)

22.8 trademgen/ui/qt/ Directory Reference

Directories

- directory [trademgen](#)

22.9 trademgen/service/ Directory Reference

Files

- file [TRADEMGEN_Service.cpp](#)
- file [TRADEMGEN_ServiceContext.cpp](#)
- file [TRADEMGEN_ServiceContext.hpp](#)

22.10 test/ Directory Reference

Directories

- directory [trademgen](#)

22.11 trademgen/ui/qt/trademgen/ Directory Reference

Files

- file [main.cpp](#)
- file [trademgen.cpp](#)

22.12 trademgen/ Directory Reference

Directories

- directory [basic](#)
- directory [batches](#)
- directory [bom](#)
- directory [command](#)
- directory [config](#)
- directory [factory](#)
- directory [python](#)
- directory [service](#)
- directory [ui](#)

Files

- file [DBParams.hpp](#)
- file [TRADEMGEN_Abstract.hpp](#)
- file [TRADEMGEN_Exceptions.hpp](#)
- file [TRADEMGEN_Service.hpp](#)
- file [TRADEMGEN_Types.hpp](#)

22.13 test/trademgen/ Directory Reference

Files

- file [DemandGenerationTestSuite.cpp](#)
- file [DemandGenerationTestSuite.hpp](#)
- file [generateEvents.cpp](#)

22.14 trademgen/ui/ Directory Reference

Directories

- directory [qt](#)

23 Namespace Documentation

23.1 stdair Namespace Reference

Forward declarations.

Classes

- struct [CategoricalAttribute](#)
Class modeling the distribution of values that can be taken by a categorical attribute.

23.1.1 Detailed Description

Forward declarations.

23.2 TRADEMGEN Namespace Reference

Namespaces

- namespace [DemandParserHelper](#)

Classes

- struct [DefaultMap](#)
- struct [CategoricalAttributeLite](#)
Class modeling the distribution of values that can be taken by a categorical attribute.
- struct [ContinuousAttribute](#)
- struct [ContinuousAttributeLite](#)
Class modeling the distribution of values that can be taken by a continuous attribute.
- struct [DemandCharacteristics](#)
Class modeling the characteristics of a demand type.
- struct [DemandDistribution](#)
Class modeling the distribution of a demand type.
- class [DictionaryManager](#)
Class wrapper of dictionary business methods.
- struct [RandomGenerationContext](#)

- struct [FlagSaver](#)
- class [BomDisplay](#)
Utility class to display TraDemGen objects with a pretty format.
- class [DemandStream](#)
Class modeling a demand stream.
- struct [DemandStreamKey](#)
- struct [DemandStruct](#)
- class [DBManager](#)
- class [DemandManager](#)
Utility class for Demand and [DemandStream](#) objects.
- class [DemandParser](#)
Class wrapping the parser entry point.
- class [DemandFileParser](#)
- struct [DBParams](#)
- class [FacTRADEMGENSEerviceContext](#)
Factory for creating the TraDemGen service context instance.
- struct [Trademgener](#)
- class [TRADEMGEN_ServiceContext](#)
Class holding the context of the Trademgen services.
- struct [TRADEMGEN_Abstract](#)
- class [TrademgenGenerationException](#)
- class [DemandInputFileNotFoundException](#)
- class [IndexOutOfRangeException](#)
- class [TRADEMGEN_Service](#)
class holding the services related to Travel Demand Generation.

Typedefs

- typedef char [char_t](#)
- typedef
boost::spirit::classic::file_iterator
< [char_t](#) > [iterator_t](#)
- typedef
boost::spirit::classic::scanner
< [iterator_t](#) > [scanner_t](#)
- typedef
boost::spirit::classic::rule
< [scanner_t](#) > [rule_t](#)
- typedef
boost::spirit::classic::int_parser
< unsigned int, 10, 1, 1 > [int1_p_t](#)
- typedef
boost::spirit::classic::uint_parser
< unsigned int, 10, 2, 2 > [uint2_p_t](#)
- typedef
boost::spirit::classic::uint_parser
< unsigned int, 10, 1, 2 > [uint1_2_p_t](#)
- typedef
boost::spirit::classic::uint_parser
< unsigned int, 10, 1, 3 > [uint1_3_p_t](#)
- typedef
boost::spirit::classic::uint_parser
< unsigned int, 10, 4, 4 > [uint4_p_t](#)

- typedef
boost::spirit::classic::uint_parser
< unsigned int, 10, 1, 4 > [uint1_4_p_t](#)
- typedef
boost::spirit::classic::chset
< [char_t](#) > [chset_t](#)
- typedef
boost::spirit::classic::impl::loop_traits
< [chset_t](#), unsigned int,
unsigned int >::type [repeat_p_t](#)
- typedef
boost::spirit::classic::bounded
< [uint2_p_t](#), unsigned int > [bounded2_p_t](#)
- typedef
boost::spirit::classic::bounded
< [uint1_2_p_t](#), unsigned int > [bounded1_2_p_t](#)
- typedef
boost::spirit::classic::bounded
< [uint1_3_p_t](#), unsigned int > [bounded1_3_p_t](#)
- typedef
boost::spirit::classic::bounded
< [uint4_p_t](#), unsigned int > [bounded4_p_t](#)
- typedef
boost::spirit::classic::bounded
< [uint1_4_p_t](#), unsigned int > [bounded1_4_p_t](#)
- typedef
[ContinuousAttributeLite](#)
< stdair::FloatDuration_T > [ContinuousFloatDuration_T](#)
- typedef
[ContinuousFloatDuration_T::ContinuousDistribution_T](#) [ArrivalPatternCumulativeDistribution_T](#)
- typedef
[CategoricalAttributeLite](#)
< stdair::AirportCode_T > [POSProbabilityMass_T](#)
- typedef
[POSProbabilityMass_T::ProbabilityMassFunction_T](#) [POSProbabilityMassFunction_T](#)
- typedef
[CategoricalAttributeLite](#)
< stdair::ChannelLabel_T > [ChannelProbabilityMass_T](#)
- typedef
[ChannelProbabilityMass_T::ProbabilityMassFunction_T](#) [ChannelProbabilityMassFunction_T](#)
- typedef
[CategoricalAttributeLite](#)
< stdair::TripType_T > [TripTypeProbabilityMass_T](#)
- typedef
[TripTypeProbabilityMass_T::ProbabilityMassFunction_T](#) [TripTypeProbabilityMassFunction_T](#)
- typedef
[CategoricalAttributeLite](#)
< stdair::DayDuration_T > [StayDurationProbabilityMass_T](#)
- typedef
[StayDurationProbabilityMass_T::ProbabilityMassFunction_T](#) [StayDurationProbabilityMassFunction_T](#)
- typedef
[CategoricalAttributeLite](#)
< stdair::FrequentFlyer_T > [FrequentFlyerProbabilityMass_T](#)
- typedef
[FrequentFlyerProbabilityMass_T::ProbabilityMassFunction_T](#) [FrequentFlyerProbabilityMassFunction_T](#)
- typedef
[ContinuousAttributeLite](#)
< stdair::IntDuration_T > [PreferredDepartureTimeCumulativeDistribution_T](#)

- typedef
PreferredDepartureTimeCumulativeDistribution_T::ContinuousDistribution_T PreferredDepartureTime-ContinuousDistribution_T
- typedef
ContinuousAttributeLite
< stdair::PriceValue_T > ValueOfTimeCumulativeDistribution_T
- typedef
ValueOfTimeCumulativeDistribution_T::ContinuousDistribution_T ValueOfTimeContinuousDistribution_T
- typedef
ContinuousAttributeLite
< stdair::RealNumber_T > CumulativeDistribution_T
- typedef
CumulativeDistribution_T::ContinuousDistribution_T FRAT5Pattern_T
- typedef stdair::Probability_T DictionaryKey_T
- typedef std::list< DemandStream * > DemandStreamList_T
- typedef std::map< const
stdair::MapKey_T, DemandStream * > DemandStreamMap_T
- typedef std::list< std::string > DBParamsNameList_T
- typedef boost::shared_ptr
< TRADEMGEN_Service > TRADEMGEN_ServicePtr_T

Functions

- stdair::BaseGenerator_T DEFAULT_BASE_GENERATOR (stdair::DEFAULT_RANDOM_SEED)
- stdair::UniformGenerator_T DEFAULT_UNIFORM_GENERATOR (DEFAULT_BASE_GENERATOR, DEFAULT_UNIFORM_REAL_DISTRIBUTION)

Variables

- const POSProbabilityMassFunction_T DEFAULT_POS_PROBALILITY_MASS
- const stdair::FloatDuration_T DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN = -1
- const FRAT5Pattern_T DEFAULT_FRAT5_PATTERN = DefaultMap::createFRAT5Pattern()
- const double DEFAULT_MAX_ADVANCE_PURCHASE = 330.0
- const stdair::UniformDistribution_T DEFAULT_UNIFORM_REAL_DISTRIBUTION
- stdair::BaseGenerator_T DEFAULT_BASE_GENERATOR
- stdair::UniformGenerator_T DEFAULT_UNIFORM_GENERATOR

23.2.1 Typedef Documentation

23.2.1.1 typedef char TRADEMGEN::char_t

Definition at line 31 of file [BasParserTypes.hpp](#).

23.2.1.2 typedef boost::spirit::classic::file_iterator<char_t> TRADEMGEN::iterator_t

Definition at line 35 of file [BasParserTypes.hpp](#).

23.2.1.3 typedef boost::spirit::classic::scanner<iterator_t> TRADEMGEN::scanner_t

Definition at line 36 of file [BasParserTypes.hpp](#).

23.2.1.4 typedef boost::spirit::classic::rule<scanner_t> TRADEMGEN::rule_t

Definition at line 37 of file [BasParserTypes.hpp](#).

23.2.1.5 `typedef boost::spirit::classic::int_parser<unsigned int, 10, 1, 1> TRADEMGEN::int1_p_t`

1-digit-integer parser

Definition at line 45 of file [BasParserTypes.hpp](#).

23.2.1.6 `typedef boost::spirit::classic::uint_parser<unsigned int, 10, 2, 2> TRADEMGEN::uint2_p_t`

2-digit-integer parser

Definition at line 48 of file [BasParserTypes.hpp](#).

23.2.1.7 `typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 2> TRADEMGEN::uint1_2_p_t`

Up-to-2-digit-integer parser

Definition at line 51 of file [BasParserTypes.hpp](#).

23.2.1.8 `typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 3> TRADEMGEN::uint1_3_p_t`

Up-to-3-digit-integer parser

Definition at line 54 of file [BasParserTypes.hpp](#).

23.2.1.9 `typedef boost::spirit::classic::uint_parser<unsigned int, 10, 4, 4> TRADEMGEN::uint4_p_t`

4-digit-integer parser

Definition at line 57 of file [BasParserTypes.hpp](#).

23.2.1.10 `typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 4> TRADEMGEN::uint1_4_p_t`

Up-to-4-digit-integer parser

Definition at line 60 of file [BasParserTypes.hpp](#).

23.2.1.11 `typedef boost::spirit::classic::chset<char_t> TRADEMGEN::chset_t`

character set

Definition at line 63 of file [BasParserTypes.hpp](#).

23.2.1.12 `typedef boost::spirit::classic::impl::loop_traits<chset_t, unsigned int, unsigned int>::type
TRADEMGEN::repeat_p_t`

(Repeating) sequence of a given number of characters: `repeat_p(min, max)`

Definition at line 69 of file [BasParserTypes.hpp](#).

23.2.1.13 `typedef boost::spirit::classic::bounded<uint2_p_t, unsigned int> TRADEMGEN::bounded2_p_t`

Bounded-number-of-integers parser

Definition at line 72 of file [BasParserTypes.hpp](#).

23.2.1.14 `typedef boost::spirit::classic::bounded<uint1_2_p_t, unsigned int> TRADEMGEN::bounded1_2_p_t`

Definition at line 73 of file [BasParserTypes.hpp](#).

23.2.1.15 `typedef boost::spirit::classic::bounded<uint1_3_p_t, unsigned int> TRADEMGEN::bounded1_3_p_t`

Definition at line 74 of file [BasParserTypes.hpp](#).

23.2.1.16 `typedef boost::spirit::classic::bounded<uint4_p_t, unsigned int> TRADEMGEN::bounded4_p_t`

Definition at line 75 of file [BasParserTypes.hpp](#).

23.2.1.17 `typedef boost::spirit::classic::bounded<uint1_4_p_t, unsigned int> TRADEMGEN::bounded1_4_p_t`

Definition at line 76 of file [BasParserTypes.hpp](#).

23.2.1.18 `typedef ContinuousAttributeLite<stdair::FloatDuration_T> TRADEMGEN::ContinuousFloatDuration_T`

Type definition for the continuous distribution of the duration (as a float number).

Definition at line 19 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.19 `typedef ContinuousFloatDuration_T::ContinuousDistribution_T TRADEMGEN::ArrivalPattern-CumulativeDistribution_T`

Type definition for the arrival pattern cumulative distribution.

Definition at line 22 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.20 `typedef CategoricalAttributeLite<stdair::AirportCode_T> TRADEMGEN::POSProbabilityMass_T`

Define the point-of-sale probability mass.

Definition at line 25 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.21 `typedef POSProbabilityMass_T::ProbabilityMassFunction_T TRADEMGEN::POSProbabilityMass-Function_T`

Define the probability mass function type of point-of-sale.

Definition at line 28 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.22 `typedef CategoricalAttributeLite<stdair::ChannelLabel_T> TRADEMGEN::ChannelProbabilityMass_T`

Define the booking channel probability mass.

Definition at line 31 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.23 `typedef ChannelProbabilityMass_T::ProbabilityMassFunction_T TRADEMGEN::Channel-ProbabilityMassFunction_T`

Define the probability mass function type of booking channel.

Definition at line 34 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.24 `typedef CategoricalAttributeLite<stdair::TripType_T> TRADEMGEN::TripTypeProbabilityMass_T`

Define the trip type probability mass.

Definition at line 37 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.25 `typedef TripTypeProbabilityMass_T::ProbabilityMassFunction_T TRADEMGEN::TripType-ProbabilityMassFunction_T`

Define the probability mass function type of trip type.

Definition at line 40 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.26 `typedef CategoricalAttributeLite<stdair::DayDuration_T> TRADEMGEN::StayDurationProbability-Mass_T`

Define the stay duration probability mass.

Definition at line 43 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.27 `typedef StayDurationProbabilityMass_T::ProbabilityMassFunction_T
TRADEMGEN::StayDurationProbabilityMassFunction_T`

Define the probability mass function type of stay duration.

Definition at line 46 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.28 `typedef CategoricalAttributeLite<stdair::FrequentFlyer_T> TRADEMGEN::FrequentFlyerProbability-
Mass_T`

Define the frequent flyer probability mass.

Definition at line 49 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.29 `typedef FrequentFlyerProbabilityMass_T::ProbabilityMassFunction_T
TRADEMGEN::FrequentFlyerProbabilityMassFunction_T`

Define the probability mass function type of frequent flyer.

Definition at line 52 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.30 `typedef ContinuousAttributeLite<stdair::IntDuration_T> TRADEMGEN::PreferredDepartureTime-
CumulativeDistribution_T`

Define the preferred departure time cumulative distribution.

Definition at line 55 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.31 `typedef PreferredDepartureTimeCumulativeDistribution_T::ContinuousDistribution_T
TRADEMGEN::PreferredDepartureTimeContinuousDistribution_T`

Define the preferred departure time continuous distribution.

Definition at line 58 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.32 `typedef ContinuousAttributeLite<stdair::PriceValue_T> TRADEMGEN::ValueOfTimeCumulative-
Distribution_T`

Define the value of time cumulative distribution.

Definition at line 61 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.33 `typedef ValueOfTimeCumulativeDistribution_T::ContinuousDistribution_T
TRADEMGEN::ValueOfTimeContinuousDistribution_T`

Define the value of time continuous distribution.

Definition at line 64 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.34 `typedef ContinuousAttributeLite<stdair::RealNumber_T> TRADEMGEN::CumulativeDistribution_T`

Define the FRAT5 pattern type.

Definition at line 67 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.35 `typedef CumulativeDistribution_T::ContinuousDistribution_T TRADEMGEN::FRAT5Pattern_T`

Definition at line 68 of file [DemandCharacteristicsTypes.hpp](#).

23.2.1.36 `typedef stdair::Probability_T TRADEMGEN::DictionaryKey_T`

Dictionary key.

Definition at line 16 of file [DictionaryManager.hpp](#).

23.2.1.37 `typedef std::list<DemandStream*> TRADEMGEN::DemandStreamList_T`

Define the airline feature list.

Definition at line 16 of file [DemandStreamTypes.hpp](#).

23.2.1.38 `typedef std::map<const stdair::MapKey_T, DemandStream*> TRADEMGEN::DemandStreamMap_T`

Define the airline feature map.

Definition at line 22 of file [DemandStreamTypes.hpp](#).

23.2.1.39 `typedef std::list<std::string> TRADEMGEN::DBParamsNameList_T`

List of names for a given (geographical) dbparams.

Definition at line 17 of file [DBParams.hpp](#).

23.2.1.40 `typedef boost::shared_ptr<TRADEMGEN_Service> TRADEMGEN::TRADEMGEN_ServicePtr_T`

(Smart) Pointer on the TraDemGen service handler.

Definition at line 15 of file [TRADEMGEN_Types.hpp](#).

23.2.2 Function Documentation

23.2.2.1 `stdair::BaseGenerator_T TRADEMGEN::DEFAULT_BASE_GENERATOR (stdair::DEFAULT_RANDOM_SEED)`

Default base generator.

23.2.2.2 `stdair::UniformGenerator_T TRADEMGEN::DEFAULT_UNIFORM_GENERATOR (DEFAULT_BASE_GENERATOR , DEFAULT_UNIFORM_REAL_DISTRIBUTION)`

Default uniform variate generator.

23.2.3 Variable Documentation

23.2.3.1 `const POSProbabilityMassFunction_T TRADEMGEN::DEFAULT_POS_PROBALILITY_MASS`

Initial value:

```
DefaultMap::createPOSProbMass()
```

Default name for the [TRADEMGEN_Service](#). Default PoS probability mass.

Default PoS probability mass.

Definition at line 16 of file [BasConst.cpp](#).

23.2.3.2 `const stdair::FloatDuration_T TRADEMGEN::DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN = -1`

Default last lower bound of daily rate interval in arrival pattern.

Definition at line 35 of file [BasConst.cpp](#).

Referenced by [TRADEMGEN::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#).

23.2.3.3 `const FRAT5Pattern_T TRADEMGEN::DEFAULT_FRAT5_PATTERN = DefaultMap::createFRAT5Pattern()`

Default FRAT5 pattern.

Definition at line 38 of file [BasConst.cpp](#).

23.2.3.4 `const double TRADEMGEN::DEFAULT_MAX_ADVANCE_PURCHASE = 330.0`

Default MAX Advance Purchase.

Definition at line 75 of file [BasConst.cpp](#).

23.2.3.5 `const stdair::UniformDistribution_T TRADEMGEN::DEFAULT_UNIFORM_REAL_DISTRIBUTION`

Default random uniform real distribution.

Definition at line 81 of file [BasConst.cpp](#).

23.2.3.6 `stdair::BaseGenerator_T TRADEMGEN::DEFAULT_BASE_GENERATOR`

Default base generator. Just here to initialise objects (e.g., `stdair::RandomGeneration`) with default generator. They are then replaced by a generator, for which the state can better be tracked/stored.

23.2.3.7 `stdair::UniformGenerator_T TRADEMGEN::DEFAULT_UNIFORM_GENERATOR`

Default uniform generator. Just here to initialise objects (e.g., `stdair::RandomGeneration`) with default generator. They are then replaced by a generator, for which the state can better be tracked/stored.

23.3 TRADEMGEN::DemandParserHelper Namespace Reference

Classes

- struct [ParserSemanticAction](#)
- struct [storePrefDepDateRangeStart](#)
- struct [storePrefDepDateRangeEnd](#)
- struct [storeDow](#)
- struct [storeOrigin](#)
- struct [storeDestination](#)
- struct [storePrefCabin](#)
- struct [storeDemandMean](#)
- struct [storeDemandStdDev](#)
- struct [storePosCode](#)
- struct [storePosProbMass](#)
- struct [storeChannelCode](#)
- struct [storeChannelProbMass](#)
- struct [storeTripCode](#)
- struct [storeTripProbMass](#)
- struct [storeStayCode](#)
- struct [storeStayProbMass](#)
- struct [storeFFCode](#)
- struct [storeFFProbMass](#)
- struct [storePrefDepTime](#)
- struct [storePrefDepTimeProbMass](#)
- struct [storeWTP](#)
- struct [storeTimeValue](#)
- struct [storeTimeValueProbMass](#)
- struct [storeDTD](#)
- struct [storeDTDProbMass](#)
- struct [doEndDemand](#)
- struct [DemandParser](#)

Functions

- [repeat_p_t airline_code_p](#) ([chset_t](#)("0-9A-Z").[derived\(\)](#), 2, 3)
- [bounded1_4_p_t flight_number_p](#) ([uint1_4_p](#).[derived\(\)](#), 0u, 9999u)
- [bounded4_p_t year_p](#) ([uint4_p](#).[derived\(\)](#), 2000u, 2099u)
- [bounded2_p_t month_p](#) ([uint2_p](#).[derived\(\)](#), 1u, 12u)
- [bounded2_p_t day_p](#) ([uint2_p](#).[derived\(\)](#), 1u, 31u)
- [repeat_p_t dow_p](#) ([chset_t](#)("0-1").[derived\(\)](#).[derived\(\)](#), 7, 7)
- [repeat_p_t airport_p](#) ([chset_t](#)("0-9A-Z").[derived\(\)](#), 3, 3)
- [bounded1_2_p_t hours_p](#) ([uint1_2_p](#).[derived\(\)](#), 0u, 23u)
- [bounded2_p_t minutes_p](#) ([uint2_p](#).[derived\(\)](#), 0u, 59u)
- [bounded2_p_t seconds_p](#) ([uint2_p](#).[derived\(\)](#), 0u, 59u)
- [chset_t cabin_code_p](#) ("A-Z")
- [chset_t passenger_type_p](#) ("A-Z")
- [chset_t ff_type_p](#) ("A-Z")
- [repeat_p_t class_code_list_p](#) ([chset_t](#)("A-Z").[derived\(\)](#), 1, 26)
- [bounded1_3_p_t stay_duration_p](#) ([uint1_3_p](#).[derived\(\)](#), 0u, 999u)

Variables

- [int1_p_t int1_p](#)
- [uint2_p_t uint2_p](#)
- [uint1_2_p_t uint1_2_p](#)
- [uint1_3_p_t uint1_3_p](#)
- [uint4_p_t uint4_p](#)
- [uint1_4_p_t uint1_4_p](#)
- [int1_p_t family_code_p](#)

23.3.1 Function Documentation

23.3.1.1 [repeat_p_t TRADEMGEN::DemandParserHelper::airline_code_p](#) ([chset_t](#)("0-9A-Z").[derived\(\)](#) , 2 , 3)

Airline Code Parser: [repeat_p](#)(2,3)[[chset_p](#)("0-9A-Z")]

23.3.1.2 [bounded1_4_p_t TRADEMGEN::DemandParserHelper::flight_number_p](#) ([uint1_4_p](#). [derived\(\)](#), 0u , 9999u)

Flight Number Parser: [limit_d](#)(0u, 9999u)[[uint1_4_p](#)]

23.3.1.3 [bounded4_p_t TRADEMGEN::DemandParserHelper::year_p](#) ([uint4_p](#). [derived\(\)](#), 2000u , 2099u)

Year Parser: [limit_d](#)(2000u, 2099u)[[uint4_p](#)]

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.1.4 [bounded2_p_t TRADEMGEN::DemandParserHelper::month_p](#) ([uint2_p](#). [derived\(\)](#), 1u , 12u)

Month Parser: [limit_d](#)(1u, 12u)[[uint2_p](#)]

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.1.5 [bounded2_p_t TRADEMGEN::DemandParserHelper::day_p](#) ([uint2_p](#). [derived\(\)](#), 1u , 31u)

Day Parser: [limit_d](#)(1u, 31u)[[uint2_p](#)]

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.1.6 **repeat_p_t** TRADEMGEN::DemandParserHelper::dow_p (chset_t("0-1").derived().derived(), 7, 7)

DOW (Day-Of-the-Week) Parser: repeat_p(7)[chset_p("0-1")]

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.1.7 **repeat_p_t** TRADEMGEN::DemandParserHelper::airport_p (chset_t("0-9A-Z").derived(), 3, 3)

Airport Parser: repeat_p(3)[chset_p("0-9A-Z")]

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.1.8 **bounded1_2_p_t** TRADEMGEN::DemandParserHelper::hours_p (uint1_2_p.derived(), 0u, 23u)

Hour Parser: limit_d(0u, 23u)[uint2_p]

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.1.9 **bounded2_p_t** TRADEMGEN::DemandParserHelper::minutes_p (uint2_p.derived(), 0u, 59u)

Minute Parser: limit_d(0u, 59u)[uint2_p]

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.1.10 **bounded2_p_t** TRADEMGEN::DemandParserHelper::seconds_p (uint2_p.derived(), 0u, 59u)

Second Parser: limit_d(0u, 59u)[uint2_p]

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.1.11 **chset_t** TRADEMGEN::DemandParserHelper::cabin_code_p ("A-Z")

Cabin code parser: chset_p("A-Z")

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.1.12 **chset_t** TRADEMGEN::DemandParserHelper::passenger_type_p ("A-Z")

Passenger type parser: chset_p("A-Z")

23.3.1.13 **chset_t** TRADEMGEN::DemandParserHelper::ff_type_p ("A-Z")

Frequent flyer type parser: chset_p("A-Z")

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.1.14 **repeat_p_t** TRADEMGEN::DemandParserHelper::class_code_list_p (chset_t("A-Z").derived(), 1, 26)

Class Code List Parser: repeat_p(1,26)[chset_p("A-Z")]

23.3.1.15 **bounded1_3_p_t** TRADEMGEN::DemandParserHelper::stay_duration_p (uint1_3_p.derived(), 0u, 999u)

Stay duration Parser: limit_d(0u, 999u)[uint3_p]

Referenced by [TRADEMGENT::DemandParserHelper::DemandParser::definition< ScannerT >::definition\(\)](#).

23.3.2 Variable Documentation

23.3.2.1 **int1_p_t** TRADEMGEN::DemandParserHelper::int1_p

1-digit-integer parser

Definition at line 450 of file [DemandParserHelper.cpp](#).

23.3.2.2 uint2_p_t TRADEMGEN::DemandParserHelper::uint2_p

2-digit-integer parser

Definition at line 453 of file [DemandParserHelper.cpp](#).

23.3.2.3 uint1_2_p_t TRADEMGEN::DemandParserHelper::uint1_2_p

Up-to-2-digit-integer parser

Definition at line 456 of file [DemandParserHelper.cpp](#).

23.3.2.4 uint1_3_p_t TRADEMGEN::DemandParserHelper::uint1_3_p

Up-to-3-digit-integer parser

Definition at line 459 of file [DemandParserHelper.cpp](#).

23.3.2.5 uint4_p_t TRADEMGEN::DemandParserHelper::uint4_p

4-digit-integer parser

Definition at line 462 of file [DemandParserHelper.cpp](#).

23.3.2.6 uint1_4_p_t TRADEMGEN::DemandParserHelper::uint1_4_p

Up-to-4-digit-integer parser

Definition at line 465 of file [DemandParserHelper.cpp](#).

23.3.2.7 int1_p_t TRADEMGEN::DemandParserHelper::family_code_p

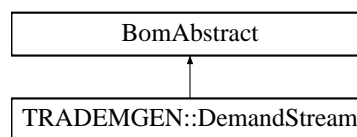
Family code parser

Definition at line 507 of file [DemandParserHelper.cpp](#).

24 Class Documentation

24.1 BomAbstract Class Reference

Inheritance diagram for BomAbstract:



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

- [trademgen/bom/DemandStream.hpp](#)

24.2 TRADEMGEN::BomDisplay Class Reference

Utility class to display TraDemGen objects with a pretty format.

```
#include <trademgen/bom/BomDisplay.hpp>
```

Static Public Member Functions

- static std::string [csvDisplay](#) (const stdair::EventQueue &)
- static void [csvDisplay](#) (std::ostream &, const [DemandStream](#) &)

24.2.1 Detailed Description

Utility class to display TraDemGen objects with a pretty format.

Definition at line 26 of file [BomDisplay.hpp](#).

24.2.2 Member Function Documentation

24.2.2.1 **std::string TRADEMGEN::BomDisplay::csvDisplay (const stdair::EventQueue & *iEventQueue*)**
[static]

Recursively display (dump in the underlying output log stream) the objects of the BOM tree.

Parameters

<i>std::ostream&</i>	Output stream in which the BOM tree should be logged/dumped.
<i>const</i>	stdair::EventQueue& Root of the BOM tree to be displayed.

Definition at line 43 of file [BomDisplay.cpp](#).

24.2.2.2 **void TRADEMGEN::BomDisplay::csvDisplay (std::ostream & *oStream*, const [DemandStream](#) & *iDemandStream*)** [static]

Recursively display (dump in the underlying output log stream) the objects of the BOM tree.

Parameters

<i>std::ostream&</i>	Output stream in which the BOM tree should be logged/dumped.
<i>const</i>	DemandStream & Root of the BOM tree to be displayed.

Definition at line 80 of file [BomDisplay.cpp](#).

References [TRADEMGEN::DemandStream::display\(\)](#).

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

- trademgen/bom/[BomDisplay.hpp](#)
- trademgen/bom/[BomDisplay.cpp](#)

24.3 stdair::CategoricalAttribute< T > Struct Template Reference

Class modeling the distribution of values that can be taken by a categorical attribute.

```
#include <trademgen/basic/CategoricalAttribute.hpp>
```

Public Types

- typedef std::map< T,
DictionaryKey_T > [ProbabilityMassFunction_T](#)
- typedef std::map
< DictionaryKey_T, T > [InverseCumulativeDistribution_T](#)

Public Member Functions

- const T & [getValue](#) (const Probability_T &iCumulativeProbability) const
- const std::string [displayProbabilityMassFunction](#) () const
- const std::string [displayInverseCumulativeDistribution](#) () const
- [CategoricalAttribute](#) (const [ProbabilityMassFunction_T](#) &iProbabilityMassFunction)
- [CategoricalAttribute](#) ()
- [CategoricalAttribute](#) (const [CategoricalAttribute](#) &iCategoricalAttribute)
- virtual [~CategoricalAttribute](#) ()
- void [determineInverseCumulativeDistributionFromProbabilityMassFunction](#) ()

24.3.1 Detailed Description

```
template<typename T>struct stdair::CategoricalAttribute< T >
```

Class modeling the distribution of values that can be taken by a categorical attribute.

Definition at line 21 of file [CategoricalAttribute.hpp](#).

24.3.2 Member Typedef Documentation

```
24.3.2.1 template<typename T > typedef std::map<T, DictionaryKey_T> stdair::CategoricalAttribute< T >::ProbabilityMassFunction_T
```

Define the probability mass function type.

Definition at line 28 of file [CategoricalAttribute.hpp](#).

```
24.3.2.2 template<typename T > typedef std::map<DictionaryKey_T, T> stdair::CategoricalAttribute< T >::InverseCumulativeDistribution_T
```

Define the inverse cumulative distribution type.

Definition at line 33 of file [CategoricalAttribute.hpp](#).

24.3.3 Constructor & Destructor Documentation

```
24.3.3.1 template<typename T > stdair::CategoricalAttribute< T >::CategoricalAttribute ( const ProbabilityMassFunction_T & iProbabilityMassFunction ) [inline]
```

Main constructor.

Definition at line 129 of file [CategoricalAttribute.hpp](#).

References [stdair::CategoricalAttribute< T >::determineInverseCumulativeDistributionFromProbabilityMassFunction\(\)](#).

```
24.3.3.2 template<typename T > stdair::CategoricalAttribute< T >::CategoricalAttribute ( ) [inline]
```

Default constructor.

Definition at line 137 of file [CategoricalAttribute.hpp](#).

```
24.3.3.3 template<typename T > stdair::CategoricalAttribute< T >::CategoricalAttribute ( const CategoricalAttribute< T > & iCategoricalAttribute ) [inline]
```

Copy constructor.

Definition at line 142 of file [CategoricalAttribute.hpp](#).

References [stdair::CategoricalAttribute< T >::determineInverseCumulativeDistributionFromProbabilityMassFunction\(\)](#).

24.3.3.4 `template<typename T> virtual stdair::CategoricalAttribute< T >::~~CategoricalAttribute ()`
`[inline, virtual]`

Destructor.

Definition at line 150 of file [CategoricalAttribute.hpp](#).

24.3.4 Member Function Documentation

24.3.4.1 `template<typename T> const T& stdair::CategoricalAttribute< T >::getValue (const Probability_T & iCumulativeProbability) const` `[inline]`

Get value from inverse cumulative distribution.

Definition at line 67 of file [CategoricalAttribute.hpp](#).

References [stdair::CategoricalAttribute< T >::displayInverseCumulativeDistribution\(\)](#).

24.3.4.2 `template<typename T> const std::string stdair::CategoricalAttribute< T >::displayProbabilityMassFunction () const` `[inline]`

Display probability mass function.

Definition at line 91 of file [CategoricalAttribute.hpp](#).

24.3.4.3 `template<typename T> const std::string stdair::CategoricalAttribute< T >::displayInverseCumulativeDistribution () const` `[inline]`

Display inverse cumulative distribution.

Definition at line 111 of file [CategoricalAttribute.hpp](#).

Referenced by [stdair::CategoricalAttribute< T >::getValue\(\)](#).

24.3.4.4 `template<typename T> void stdair::CategoricalAttribute< T >::determineInverseCumulativeDistributionFromProbabilityMassFunction ()` `[inline]`

Determine inverse cumulative distribution from probability mass function (initialisation).

Definition at line 157 of file [CategoricalAttribute.hpp](#).

Referenced by [stdair::CategoricalAttribute< T >::CategoricalAttribute\(\)](#).

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

- [trademgen/basic/CategoricalAttribute.hpp](#)

24.4 TRADEMGEN::CategoricalAttributeLite< T > Struct Template Reference

Class modeling the distribution of values that can be taken by a categorical attribute.

`#include <trademgen/basic/CategoricalAttributeLite.hpp>`

Public Types

- `typedef std::map< T, stdair::Probability_T > ProbabilityMassFunction_T`

Public Member Functions

- const T & [getValue](#) (const stdair::Probability_T &iCumulativeProbability) const
- bool [checkValue](#) (const T &iValue) const
- const std::string [displayProbabilityMass](#) () const
- [CategoricalAttributeLite](#) (const [ProbabilityMassFunction_T](#) &iValueMap)
- [CategoricalAttributeLite](#) ()
- [CategoricalAttributeLite](#) (const [CategoricalAttributeLite](#) &iCAL)
- [CategoricalAttributeLite](#) & [operator=](#) (const [CategoricalAttributeLite](#) &iCAL)
- virtual [~CategoricalAttributeLite](#) ()

24.4.1 Detailed Description

```
template<typename T>struct TRADEMGEN::CategoricalAttributeLite< T >
```

Class modeling the distribution of values that can be taken by a categorical attribute.

Definition at line 27 of file [CategoricalAttributeLite.hpp](#).

24.4.2 Member Typedef Documentation

24.4.2.1 `template<typename T> typedef std::map<T, stdair::Probability_T> TRADEMGEN::CategoricalAttributeLite< T >::ProbabilityMassFunction_T`

Type for the probability mass function.

Definition at line 33 of file [CategoricalAttributeLite.hpp](#).

24.4.3 Constructor & Destructor Documentation

24.4.3.1 `template<typename T> TRADEMGEN::CategoricalAttributeLite< T >::CategoricalAttributeLite (const ProbabilityMassFunction_T & iValueMap) [inline]`

Main constructor.

Definition at line 95 of file [CategoricalAttributeLite.hpp](#).

24.4.3.2 `template<typename T> TRADEMGEN::CategoricalAttributeLite< T >::CategoricalAttributeLite () [inline]`

Default constructor.

Definition at line 103 of file [CategoricalAttributeLite.hpp](#).

24.4.3.3 `template<typename T> TRADEMGEN::CategoricalAttributeLite< T >::CategoricalAttributeLite (const CategoricalAttributeLite< T > & iCAL) [inline]`

Copy constructor.

Definition at line 109 of file [CategoricalAttributeLite.hpp](#).

24.4.3.4 `template<typename T> virtual TRADEMGEN::CategoricalAttributeLite< T >::~~CategoricalAttributeLite () [inline, virtual]`

Destructor.

Definition at line 128 of file [CategoricalAttributeLite.hpp](#).

24.4.4 Member Function Documentation

24.4.4.1 `template<typename T> const T& TRADEMGEN::CategoricalAttributeLite< T >::getValue (const stdair::Probability_T & iCumulativeProbability) const [inline]`

Get value from inverse cumulative distribution.

Definition at line 41 of file [CategoricalAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateChannel\(\)](#), [TRADEMGENT::DemandStream::generateFrequentFlyer\(\)](#), [TRADEMGENT::DemandStream::generateStayDuration\(\)](#), [TRADEMGENT::DemandStream::generateTripType\(\)](#), and [TRADEMGENT::DemandCharacteristics::getPOSValue\(\)](#).

24.4.4.2 `template<typename T> bool TRADEMGENT::CategoricalAttributeLite< T >::checkValue (const T & iValue) const [inline]`

Check if a value belongs to the value list.

Definition at line 61 of file [CategoricalAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandCharacteristics::checkPOSValue\(\)](#).

24.4.4.3 `template<typename T> const std::string TRADEMGENT::CategoricalAttributeLite< T >::displayProbabilityMass () const [inline]`

Display probability mass function.

Definition at line 76 of file [CategoricalAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandCharacteristics::describe\(\)](#), [TRADEMGENT::DemandStream::display\(\)](#), and [TRADEMGENT::CategoricalAttributeLite< stdair::TripType_T >::getValue\(\)](#).

24.4.4.4 `template<typename T> CategoricalAttributeLite& TRADEMGENT::CategoricalAttributeLite< T >::operator= (const CategoricalAttributeLite< T > & iCAL) [inline]`

Copy operator.

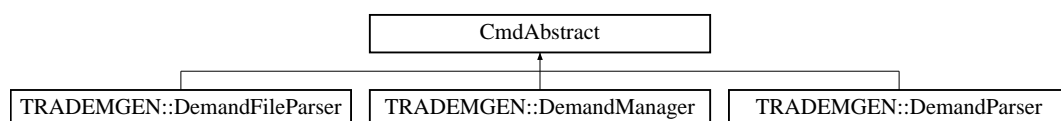
Definition at line 118 of file [CategoricalAttributeLite.hpp](#).

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

- [trademgen/basic/CategoricalAttributeLite.hpp](#)

24.5 CmdAbstract Class Reference

Inheritance diagram for CmdAbstract:



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

- [trademgen/command/DemandParser.hpp](#)

24.6 TRADEMGENT::ContinuousAttribute< T > Struct Template Reference

```
#include <trademgen/basic/ContinuousAttribute.hpp>
```

Public Types

- typedef std::multimap< T, [DictionaryKey_T](#) > [ContinuousDistribution_T](#)
- typedef std::multimap< [DictionaryKey_T](#), T > [ContinuousInverseDistribution_T](#)

Public Member Functions

- const T [getValue](#) (const stdair::Probability_T &iCumulativeProbability) const
- const std::string [displayCumulativeDistribution](#) () const
- const std::string [displayInverseCumulativeDistribution](#) () const
- [ContinuousAttribute](#) ()
- [ContinuousAttribute](#) (const [ContinuousDistribution_T](#) &iCumulativeDistribution)
- [ContinuousAttribute](#) (const [ContinuousAttribute](#) &iContinuousAttribute)
- virtual [~ContinuousAttribute](#) ()
- void [determineInverseCumulativeDistributionFromCumulativeDistribution](#) ()

24.6.1 Detailed Description

template<class T>struct TRADEMGEN::ContinuousAttribute< T >

Class modeling the distribution of values that can be taken by a continuous attribute.

Definition at line 21 of file [ContinuousAttribute.hpp](#).

24.6.2 Member Typedef Documentation

24.6.2.1 template<class T > typedef std::multimap<T, [DictionaryKey_T](#)> [TRADEMGENT::ContinuousAttribute< T >::ContinuousDistribution_T](#)

Definition at line 26 of file [ContinuousAttribute.hpp](#).

24.6.2.2 template<class T > typedef std::multimap<[DictionaryKey_T](#), T> [TRADEMGENT::ContinuousAttribute< T >::ContinuousInverseDistribution_T](#)

Definition at line 27 of file [ContinuousAttribute.hpp](#).

24.6.3 Constructor & Destructor Documentation

24.6.3.1 template<class T > [TRADEMGENT::ContinuousAttribute< T >::ContinuousAttribute](#) () [\[inline\]](#)

Constructor by default

Definition at line 113 of file [ContinuousAttribute.hpp](#).

24.6.3.2 template<class T > [TRADEMGENT::ContinuousAttribute< T >::ContinuousAttribute](#) (const [ContinuousDistribution_T](#) & iCumulativeDistribution) [\[inline\]](#)

Constructor

Definition at line 116 of file [ContinuousAttribute.hpp](#).

References [TRADEMGENT::ContinuousAttribute< T >::determineInverseCumulativeDistributionFromCumulativeDistribution\(\)](#).

24.6.3.3 `template<class T > TRADEMGEN::ContinuousAttribute< T >::ContinuousAttribute (const ContinuousAttribute< T > & iContinuousAttribute) [inline]`

Copy constructor

Definition at line 122 of file [ContinuousAttribute.hpp](#).

24.6.3.4 `template<class T > virtual TRADEMGEN::ContinuousAttribute< T >::~~ContinuousAttribute () [inline, virtual]`

Destructor

Definition at line 128 of file [ContinuousAttribute.hpp](#).

24.6.4 Member Function Documentation

24.6.4.1 `template<class T > const T TRADEMGEN::ContinuousAttribute< T >::getValue (const stdair::Probability.T & iCumulativeProbability) const [inline]`

Get value from inverse cumulative distribution.

Definition at line 52 of file [ContinuousAttribute.hpp](#).

References [TRADEMGENT::DictionaryManager::keyToValue\(\)](#), and [TRADEMGENT::DictionaryManager::valueToKey\(\)](#).

24.6.4.2 `template<class T > const std::string TRADEMGEN::ContinuousAttribute< T >::displayCumulativeDistribution () const [inline]`

Display cumulative distribution

Definition at line 83 of file [ContinuousAttribute.hpp](#).

References [TRADEMGENT::DictionaryManager::keyToValue\(\)](#).

24.6.4.3 `template<class T > const std::string TRADEMGEN::ContinuousAttribute< T >::displayInverseCumulativeDistribution () const [inline]`

Display inverse cumulative distribution

Definition at line 99 of file [ContinuousAttribute.hpp](#).

References [TRADEMGENT::DictionaryManager::keyToValue\(\)](#).

24.6.4.4 `template<class T > void TRADEMGEN::ContinuousAttribute< T >::determineInverseCumulativeDistributionFromCumulativeDistribution () [inline]`

Determine inverse cumulative distribution from cumulative distribution (initialisation).

Definition at line 132 of file [ContinuousAttribute.hpp](#).

Referenced by [TRADEMGENT::ContinuousAttribute< T >::ContinuousAttribute\(\)](#).

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

- [trademgen/basic/ContinuousAttribute.hpp](#)

24.7 TRADEMGEN::ContinuousAttributeLite< T > Struct Template Reference

Class modeling the distribution of values that can be taken by a continuous attribute.

```
#include <trademgen/basic/ContinuousAttributeLite.hpp>
```

Public Types

- typedef std::map< T, stdair::Probability_T > [ContinuousDistribution_T](#)

Public Member Functions

- const T [getValue](#) (const stdair::Probability_T &iCumulativeProbability) const
- const double [getDerivativeValue](#) (const T iKey) const
- const T [getUpperBound](#) (const T iKey) const
- const std::string [displayCumulativeDistribution](#) () const
- [ContinuousAttributeLite](#) (const [ContinuousDistribution_T](#) &iValueMap)
- [ContinuousAttributeLite](#) (const [ContinuousAttributeLite](#) &iCAL)
- [ContinuousAttributeLite](#) & [operator=](#) (const [ContinuousAttributeLite](#) &iCAL)
- virtual [~ContinuousAttributeLite](#) ()

24.7.1 Detailed Description

template<typename T>struct TRADEMGEN::ContinuousAttributeLite< T >

Class modeling the distribution of values that can be taken by a continuous attribute.

Definition at line 26 of file [ContinuousAttributeLite.hpp](#).

24.7.2 Member Typedef Documentation

24.7.2.1 template<typename T> typedef std::map<T, stdair::Probability_T> TRADEMGEN::ContinuousAttributeLite< T >::ContinuousDistribution_T

Type for the probability mass function.

Definition at line 32 of file [ContinuousAttributeLite.hpp](#).

24.7.3 Constructor & Destructor Documentation

24.7.3.1 template<typename T> TRADEMGEN::ContinuousAttributeLite< T >::ContinuousAttributeLite (const ContinuousDistribution_T & iValueMap) [inline]

Constructor.

Definition at line 157 of file [ContinuousAttributeLite.hpp](#).

24.7.3.2 template<typename T> TRADEMGEN::ContinuousAttributeLite< T >::ContinuousAttributeLite (const ContinuousAttributeLite< T > & iCAL) [inline]

Copy constructor.

Definition at line 165 of file [ContinuousAttributeLite.hpp](#).

24.7.3.3 template<typename T> virtual TRADEMGEN::ContinuousAttributeLite< T >::~~ContinuousAttributeLite () [inline, virtual]

Destructor.

Definition at line 184 of file [ContinuousAttributeLite.hpp](#).

24.7.4 Member Function Documentation

24.7.4.1 `template<typename T> const T TRADEMGEN::ContinuousAttributeLite< T >::getValue (const stdair::Probability_T & iCumulativeProbability) const [inline]`

Get value from inverse cumulative distribution.

Definition at line 39 of file [ContinuousAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#), [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#), [TRADEMGENT::DemandStream::generateValueOfTime\(\)](#), and [TRADEMGENT::DemandStream::generateWTP\(\)](#).

24.7.4.2 `template<typename T> const double TRADEMGEN::ContinuousAttributeLite< T >::getDerivativeValue (const T iKey) const [inline]`

Get the value of the derivative function in a key point.

Definition at line 82 of file [ContinuousAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#).

24.7.4.3 `template<typename T> const T TRADEMGEN::ContinuousAttributeLite< T >::getUpperBound (const T iKey) const [inline]`

Get the upper bound.

Definition at line 116 of file [ContinuousAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#).

24.7.4.4 `template<typename T> const std::string TRADEMGEN::ContinuousAttributeLite< T >::displayCumulativeDistribution () const [inline]`

Display cumulative distribution.

Definition at line 135 of file [ContinuousAttributeLite.hpp](#).

Referenced by [TRADEMGENT::DemandCharacteristics::describe\(\)](#).

24.7.4.5 `template<typename T> ContinuousAttributeLite& TRADEMGEN::ContinuousAttributeLite< T >::operator= (const ContinuousAttributeLite< T > & iCAL) [inline]`

Copy operator.

Definition at line 174 of file [ContinuousAttributeLite.hpp](#).

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

- [trademgen/basic/ContinuousAttributeLite.hpp](#)

24.8 TRADEMGEN::DBManager Class Reference

```
#include <trademgen/command/DBManager.hpp>
```

Static Public Member Functions

- static void [updateAirlineInDB](#) (stdair::DBSession_T &, const stdair::AirlineStruct &)
- static bool [retrieveAirline](#) (stdair::DBSession_T &, const stdair::AirlineCode_T &, stdair::AirlineStruct &)
- static void [prepareSelectStatement](#) (stdair::DBSession_T &, stdair::DBRequestStatement_T &, stdair::AirlineStruct &)
- static bool [iterateOnStatement](#) (stdair::DBRequestStatement_T &, stdair::AirlineStruct &, const bool iShouldDoReset)

24.8.1 Detailed Description

Class building the Business Object Model (BOM) from data retrieved from the database.

Definition at line 20 of file [DBManager.hpp](#).

24.8.2 Member Function Documentation

24.8.2.1 void **TRADEMGEN::DBManager::updateAirlineInDB** (*stdair::DBSession_T* & *ioSociSession*, const *stdair::AirlineStruct* & *iAirline*) [static]

Update the fields of the database row corresponding to the given BOM object.

Definition at line 121 of file [DBManager.cpp](#).

24.8.2.2 bool **TRADEMGEN::DBManager::retrieveAirline** (*stdair::DBSession_T* & *ioSociSession*, const *stdair::AirlineCode_T* & *iAirlineCode*, *stdair::AirlineStruct* & *ioAirline*) [static]

Retrieve, from the (MySQL) database, the row corresponding to the given BOM code, and fill the given BOM object with that retrieved data.

Definition at line 157 of file [DBManager.cpp](#).

References [iterateOnStatement\(\)](#).

24.8.2.3 void **TRADEMGEN::DBManager::prepareSelectStatement** (*stdair::DBSession_T* & *ioSociSession*, *stdair::DBRequestStatement_T* & *ioSelectStatement*, *stdair::AirlineStruct* & *ioAirline*) [static]

Prepare (parse and put in cache) the SQL statement.

Definition at line 24 of file [DBManager.cpp](#).

24.8.2.4 bool **TRADEMGEN::DBManager::iterateOnStatement** (*stdair::DBRequestStatement_T* & *ioStatement*, *stdair::AirlineStruct* & *ioAirline*, const bool *iShouldDoReset*) [static]

Iterate on the SQL statement.

The SQL has to be already prepared. const bool Tells whether the Airline object should be reset.

Definition at line 97 of file [DBManager.cpp](#).

Referenced by [retrieveAirline\(\)](#).

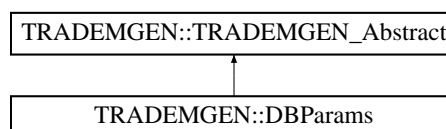
The documentation for this class was generated from the following files:

- [trademgen/command/DBManager.hpp](#)
- [trademgen/command/DBManager.cpp](#)

24.9 TRADEMGEN::DBParams Struct Reference

```
#include <trademgen/DBParams.hpp>
```

Inheritance diagram for TRADEMGEN::DBParams:



Public Member Functions

- `std::string` [getUser](#) () const
- `std::string` [getPassword](#) () const
- `std::string` [getHost](#) () const
- `std::string` [getPort](#) () const
- `std::string` [getDBName](#) () const
- `void` [setUser](#) (const `std::string` &iUser)
- `void` [setPassword](#) (const `std::string` &iPasswd)
- `void` [setHost](#) (const `std::string` &iHost)
- `void` [setPort](#) (const `std::string` &iPort)
- `void` [setDBName](#) (const `std::string` &iDBName)
- `bool` [check](#) () const
- `void` [toStream](#) (`std::ostream` &ioOut) const
- `void` [fromStream](#) (`std::istream` &)
- `std::string` [toShortString](#) () const
- `std::string` [toString](#) () const
- [DBParams](#) (const `std::string` &iDBUser, const `std::string` &iDBPasswd, const `std::string` &iDBHost, const `std::string` &iDBPort, const `std::string` &iDBName)
- `virtual` [~DBParams](#) ()

24.9.1 Detailed Description

Structure modelling a (geographical) dbparams.

Definition at line 21 of file [DBParams.hpp](#).

24.9.2 Constructor & Destructor Documentation

24.9.2.1 TRADEMGEN::DBParams::DBParams (const `std::string` &iDBUser, const `std::string` &iDBPasswd, const `std::string` &iDBHost, const `std::string` &iDBPort, const `std::string` &iDBName) `[inline]`

Main Constructor.

Definition at line 119 of file [DBParams.hpp](#).

24.9.2.2 virtual TRADEMGEN::DBParams::~~DBParams () `[inline, virtual]`

Default Constructor. Default copy constructor. Destructor.

Definition at line 132 of file [DBParams.hpp](#).

24.9.3 Member Function Documentation

24.9.3.1 std::string TRADEMGEN::DBParams::getUser () const `[inline]`

Get the database user name.

Definition at line 25 of file [DBParams.hpp](#).

24.9.3.2 std::string TRADEMGEN::DBParams::getPassword () const `[inline]`

Get the database user password.

Definition at line 30 of file [DBParams.hpp](#).

24.9.3.3 `std::string TRADEMGEN::DBParams::getHost () const` `[inline]`

Get the database host name.

Definition at line 35 of file [DBParams.hpp](#).

24.9.3.4 `std::string TRADEMGEN::DBParams::getPort () const` `[inline]`

Get the database port number.

Definition at line 40 of file [DBParams.hpp](#).

24.9.3.5 `std::string TRADEMGEN::DBParams::getDBName () const` `[inline]`

Get the database name.

Definition at line 45 of file [DBParams.hpp](#).

24.9.3.6 `void TRADEMGEN::DBParams::setUser (const std::string & iUser)` `[inline]`

Set the database user name.

Definition at line 52 of file [DBParams.hpp](#).

24.9.3.7 `void TRADEMGEN::DBParams::setPassword (const std::string & iPasswd)` `[inline]`

Set the database password.

Definition at line 57 of file [DBParams.hpp](#).

24.9.3.8 `void TRADEMGEN::DBParams::setHost (const std::string & iHost)` `[inline]`

Set the database host name.

Definition at line 62 of file [DBParams.hpp](#).

24.9.3.9 `void TRADEMGEN::DBParams::setPort (const std::string & iPort)` `[inline]`

Set the database port number.

Definition at line 67 of file [DBParams.hpp](#).

24.9.3.10 `void TRADEMGEN::DBParams::setDBName (const std::string & iDBName)` `[inline]`

Set the database name.

Definition at line 72 of file [DBParams.hpp](#).

24.9.3.11 `bool TRADEMGEN::DBParams::check () const` `[inline]`

Check that all the parameters are fine.

Definition at line 80 of file [DBParams.hpp](#).

24.9.3.12 `void TRADEMGEN::DBParams::toStream (std::ostream & ioOut) const` `[inline, virtual]`

Dump a structure into an output stream.

Parameters

<i>ostream&</i>	the output stream.
---------------------	--------------------

Implements [TRADEMGENTRADEMGENTAbstract](#).

Definition at line 93 of file [DBParams.hpp](#).

References [toString\(\)](#).

24.9.3.13 `void TRADEMGEN::DBParams::fromStream (std::istream &) [inline, virtual]`

Read a structure from an input stream.

Parameters

<i>istream&</i>	the input stream.
---------------------	-------------------

Implements [TRADEMGENTRADEMGENTAbstract](#).

Definition at line 99 of file [DBParams.hpp](#).

24.9.3.14 `std::string TRADEMGEN::DBParams::toShortString () const [inline]`

Get a short display of the [DBParams](#) structure.

Definition at line 103 of file [DBParams.hpp](#).

24.9.3.15 `std::string TRADEMGEN::DBParams::toString () const [inline, virtual]`

Get the serialised version of the [DBParams](#) structure.

Implements [TRADEMGENTRADEMGENTAbstract](#).

Definition at line 110 of file [DBParams.hpp](#).

Referenced by [toStream\(\)](#).

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

- [trademgen/DBParams.hpp](#)

24.10 TRADEMGEN::DefaultMap Struct Reference

```
#include <trademgen/basic/BasConst_DemandGeneration.hpp>
```

Static Public Member Functions

- static [POSProbabilityMassFunction_T](#) [createPOSProbMass](#) ()
- static [FRAT5Pattern_T](#) [createFRAT5Pattern](#) ()

24.10.1 Detailed Description

Default PoS probability mass.

Definition at line 24 of file [BasConst_DemandGeneration.hpp](#).

24.10.2 Member Function Documentation

24.10.2.1 `POSProbabilityMassFunction_T TRADEMGEN::DefaultMap::createPOSProbMass () [static]`

Default PoS probability mass.

Definition at line 20 of file [BasConst.cpp](#).

24.10.2.2 `FRAT5Pattern_T TRADEMGEN::DefaultMap::createFRAT5Pattern () [static]`

Default FRAT5 pattern.

Definition at line 41 of file [BasConst.cpp](#).

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

- trademgen/basic/[BasConst_DemandGeneration.hpp](#)
- trademgen/basic/[BasConst.cpp](#)

24.11 TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT > Struct Template Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Public Member Functions

- [definition](#) ([DemandParser](#) const &self)
- boost::spirit::classic::rule
< ScannerT > const & [start](#) () const

Public Attributes

- boost::spirit::classic::rule
< ScannerT > [demand_list](#)
- boost::spirit::classic::rule
< ScannerT > [not_to_be_parsed](#)
- boost::spirit::classic::rule
< ScannerT > [demand](#)
- boost::spirit::classic::rule
< ScannerT > [demand_end](#)
- boost::spirit::classic::rule
< ScannerT > [pref_dep_date_range](#)
- boost::spirit::classic::rule
< ScannerT > [date](#)
- boost::spirit::classic::rule
< ScannerT > [dow](#)
- boost::spirit::classic::rule
< ScannerT > [origin](#)
- boost::spirit::classic::rule
< ScannerT > [destination](#)
- boost::spirit::classic::rule
< ScannerT > [pref_cabin](#)
- boost::spirit::classic::rule
< ScannerT > [demand_params](#)
- boost::spirit::classic::rule
< ScannerT > [pos_dist](#)
- boost::spirit::classic::rule
< ScannerT > [pos_pair](#)
- boost::spirit::classic::rule
< ScannerT > [pos_code](#)
- boost::spirit::classic::rule
< ScannerT > [pos_share](#)
- boost::spirit::classic::rule
< ScannerT > [channel_dist](#)
- boost::spirit::classic::rule
< ScannerT > [channel_pair](#)
- boost::spirit::classic::rule
< ScannerT > [channel_code](#)

- boost::spirit::classic::rule
 < ScannerT > [channel_share](#)
- boost::spirit::classic::rule
 < ScannerT > [trip_dist](#)
- boost::spirit::classic::rule
 < ScannerT > [trip_pair](#)
- boost::spirit::classic::rule
 < ScannerT > [trip_code](#)
- boost::spirit::classic::rule
 < ScannerT > [trip_share](#)
- boost::spirit::classic::rule
 < ScannerT > [stay_dist](#)
- boost::spirit::classic::rule
 < ScannerT > [stay_pair](#)
- boost::spirit::classic::rule
 < ScannerT > [stay_share](#)
- boost::spirit::classic::rule
 < ScannerT > [ff_dist](#)
- boost::spirit::classic::rule
 < ScannerT > [ff_pair](#)
- boost::spirit::classic::rule
 < ScannerT > [ff_code](#)
- boost::spirit::classic::rule
 < ScannerT > [ff_share](#)
- boost::spirit::classic::rule
 < ScannerT > [pref_dep_time_dist](#)
- boost::spirit::classic::rule
 < ScannerT > [pref_dep_time_pair](#)
- boost::spirit::classic::rule
 < ScannerT > [pref_dep_time_share](#)
- boost::spirit::classic::rule
 < ScannerT > [time](#)
- boost::spirit::classic::rule
 < ScannerT > [wtp](#)
- boost::spirit::classic::rule
 < ScannerT > [time_value_dist](#)
- boost::spirit::classic::rule
 < ScannerT > [time_value_pair](#)
- boost::spirit::classic::rule
 < ScannerT > [time_value_share](#)
- boost::spirit::classic::rule
 < ScannerT > [dtd_dist](#)
- boost::spirit::classic::rule
 < ScannerT > [dtd_pair](#)
- boost::spirit::classic::rule
 < ScannerT > [dtd_share](#)

24.11.1 Detailed Description

template<typename ScannerT>struct TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >

Definition at line 352 of file [DemandParserHelper.hpp](#).

24.11.2 Constructor & Destructor Documentation

24.11.2.1 `template<typename ScannerT > TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::definition (DemandParser const & self)`

Definition at line 532 of file [DemandParserHelper.cpp](#).

References [TRADEMG-EN::DemandParserHelper::airport_p\(\)](#), [TRADEMG-EN::DemandParserHelper::cabin_code_p\(\)](#), [TRADEMG-EN::DemandParserHelper::day_p\(\)](#), [TRADEMG-EN::DemandParserHelper::dow_p\(\)](#), [TRADEMG-EN::DemandParserHelper::ff_type_p\(\)](#), [TRADEMG-EN::DemandParserHelper::hours_p\(\)](#), [TRADEMG-EN::DemandParserHelper::minutes_p\(\)](#), [TRADEMG-EN::DemandParserHelper::month_p\(\)](#), [TRADEMG-EN::DemandParserHelper::seconds_p\(\)](#), [TRADEMG-EN::DemandParserHelper::stay_duration_p\(\)](#), and [TRADEMG-EN::DemandParserHelper::year_p\(\)](#).

24.11.3 Member Function Documentation

24.11.3.1 `template<typename ScannerT > bsc::rule< ScannerT > const & TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::start () const`

Entry point of the parser.

Definition at line 776 of file [DemandParserHelper.cpp](#).

24.11.4 Member Data Documentation

24.11.4.1 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::demand_list`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.2 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::not_to_be_parsed`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.3 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::demand`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.4 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::demand_end`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.5 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pref_dep_date_range`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.6 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::date`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.7 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::dow`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.8 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::origin`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.9 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::destination`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.10 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pref_cabin`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.11 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::demand_params`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.12 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pos_dist`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.13 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pos_pair`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.14 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pos_code`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.15 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pos_share`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.16 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::channel_dist`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.17 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::channel_pair`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.18 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::channel_code`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.19 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::channel_share`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.20 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::trip_dist`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.21 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::trip_pair`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.22 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::trip_code`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.23 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::trip_share`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.24 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::stay_dist`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.25 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::stay_pair`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.26 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::stay_share`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.27 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::ff_dist`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.28 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::ff_pair`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.29 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::ff_code`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.30 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::ff_share`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.31 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pref_dep_time_dist`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.32 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pref_dep_time_pair`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.33 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::pref_dep_time_share`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.34 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::time`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.35 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::wtp`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.36 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::time_value_dist`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.37 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::time_value_pair`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.38 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::time_value_share`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.39 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::dtd_dist`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.40 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::dtd_pair`

Definition at line 356 of file [DemandParserHelper.hpp](#).

24.11.4.41 `template<typename ScannerT > boost::spirit::classic::rule<ScannerT> TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >::dtd_share`

Definition at line 356 of file [DemandParserHelper.hpp](#).

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

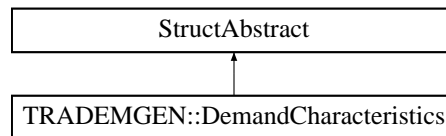
- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.12 TRADEMGEN::DemandCharacteristics Struct Reference

Class modeling the characteristics of a demand type.

```
#include <trademgen/basic/DemandCharacteristics.hpp>
```

Inheritance diagram for TRADEMGEN::DemandCharacteristics:



Public Member Functions

- `const stdair::AirportCode_T & getPOSValue (const stdair::Probability_T & iCumulativeProbability) const`
- `bool checkPOSValue (const stdair::AirportCode_T & iPOS) const`
- `const std::string describe () const`
- `DemandCharacteristics (const ArrivalPatternCumulativeDistribution_T &, const POSProbabilityMassFunction_T &, const ChannelProbabilityMassFunction_T &, const TripTypeProbabilityMassFunction_T &, const StayDurationProbabilityMassFunction_T &, const FrequentFlyerProbabilityMassFunction_T &, const PreferredDepartureTimeContinuousDistribution_T &, const stdair::WTP_T &, const ValueOfTimeContinuousDistribution_T &)`
- `DemandCharacteristics ()`
- `DemandCharacteristics (const DemandCharacteristics &)`
- `~DemandCharacteristics ()`

Public Attributes

- `ContinuousFloatDuration_T _arrivalPattern`
- `POSProbabilityMass_T _posProbabilityMass`
- `ChannelProbabilityMass_T _channelProbabilityMass`
- `TripTypeProbabilityMass_T _tripTypeProbabilityMass`
- `StayDurationProbabilityMass_T _stayDurationProbabilityMass`
- `FrequentFlyerProbabilityMass_T _frequentFlyerProbabilityMass`
- `PreferredDepartureTimeCumulativeDistribution_T _preferredDepartureTimeCumulativeDistribution`
- `stdair::WTP_T _minWTP`
- `CumulativeDistribution_T _frat5Pattern`
- `ValueOfTimeCumulativeDistribution_T _valueOfTimeCumulativeDistribution`

24.12.1 Detailed Description

Class modeling the characteristics of a demand type.

Definition at line 21 of file [DemandCharacteristics.hpp](#).

24.12.2 Constructor & Destructor Documentation

- 24.12.2.1 TRADEMGEN::DemandCharacteristics::DemandCharacteristics (const ArrivalPatternCumulativeDistribution_T & iArrivalPattern, const POSProbabilityMassFunction_T & iPOSProbMass, const ChannelProbabilityMassFunction_T & iChannelProbMass, const TripTypeProbabilityMassFunction_T & iTripTypeProbMass, const StayDurationProbabilityMassFunction_T & iStayDurationProbMass, const FrequentFlyerProbabilityMassFunction_T & iFrequentFlyerProbMass, const PreferredDepartureTimeContinuousDistribution_T & iPreferredDepartureTimeContinuousDistribution, const stdair::WTP_T & iMinWTP, const ValueOfTimeContinuousDistribution_T & iValueOfTimeContinuousDistribution)**

Constructor.

Definition at line 44 of file [DemandCharacteristics.cpp](#).

24.12.2.2 TRADEMGEN::DemandCharacteristics::DemandCharacteristics ()

Default constructor.

Definition at line 16 of file [DemandCharacteristics.cpp](#).

24.12.2.3 TRADEMGEN::DemandCharacteristics::DemandCharacteristics (const DemandCharacteristics & *idC*)

Copy constructor.

Definition at line 30 of file [DemandCharacteristics.cpp](#).

24.12.2.4 TRADEMGEN::DemandCharacteristics::~~DemandCharacteristics ()

Destructor.

Definition at line 65 of file [DemandCharacteristics.cpp](#).

24.12.3 Member Function Documentation**24.12.3.1 const stdair::AirportCode_T & TRADEMGEN::DemandCharacteristics::getPOSValue (const stdair::Probability_T & *iCumulativeProbability*) const**

Get the POS corresponding to the cumulative probability

Definition at line 70 of file [DemandCharacteristics.cpp](#).

References [_posProbabilityMass](#), and [TRADEMGENT::CategoricalAttributeLite< T >::getValue\(\)](#).

Referenced by [TRADEMGENT::DemandStream::generatePOS\(\)](#).

24.12.3.2 bool TRADEMGEN::DemandCharacteristics::checkPOSValue (const stdair::AirportCode_T & *iPOS*) const

Check that the POS is within the distribution.

Definition at line 76 of file [DemandCharacteristics.cpp](#).

References [_posProbabilityMass](#), and [TRADEMGENT::CategoricalAttributeLite< T >::checkValue\(\)](#).

24.12.3.3 const std::string TRADEMGEN::DemandCharacteristics::describe () const

Give a description of the structure (for display purposes).

Definition at line 81 of file [DemandCharacteristics.cpp](#).

References [_arrivalPattern](#), [_channelProbabilityMass](#), [_frequentFlyerProbabilityMass](#), [_minWTP](#), [_posProbabilityMass](#), [_preferredDepartureTimeCumulativeDistribution](#), [_stayDurationProbabilityMass](#), [_tripTypeProbabilityMass](#), [_valueOfTimeCumulativeDistribution](#), [TRADEMGENT::ContinuousAttributeLite< T >::displayCumulativeDistribution\(\)](#), and [TRADEMGENT::CategoricalAttributeLite< T >::displayProbabilityMass\(\)](#).

Referenced by [TRADEMGENT::DemandStream::display\(\)](#).

24.12.4 Member Data Documentation**24.12.4.1 ContinuousFloatDuration_T TRADEMGEN::DemandCharacteristics::_arrivalPattern**

Arrival pattern (cumulative distribution of timing of arrival of requests (negative number of days between departure date and request date)).

Definition at line 83 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGENT::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#), and [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#).

24.12.4.2 POSProbabilityMass_T TRADEMGEN::DemandCharacteristics::_posProbabilityMass

POS probability mass.

Definition at line 88 of file [DemandCharacteristics.hpp](#).

Referenced by [checkPOSValue\(\)](#), [describe\(\)](#), and [getPOSValue\(\)](#).

24.12.4.3 ChannelProbabilityMass_T TRADEMGEN::DemandCharacteristics::_channelProbabilityMass

Channel probability mass.

Definition at line 93 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandStream::generateChannel\(\)](#).

24.12.4.4 TripTypeProbabilityMass_T TRADEMGEN::DemandCharacteristics::_tripTypeProbabilityMass

Trip type probability mass.

Definition at line 98 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandStream::generateTripType\(\)](#).

24.12.4.5 StayDurationProbabilityMass_T TRADEMGEN::DemandCharacteristics::_stayDuration-ProbabilityMass

Stay duration probability mass.

Definition at line 103 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandStream::generateStayDuration\(\)](#).

24.12.4.6 FrequentFlyerProbabilityMass_T TRADEMGEN::DemandCharacteristics::_frequentFlyer-ProbabilityMass

Frequent flyer probability mass.

Definition at line 108 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandStream::generateFrequentFlyer\(\)](#).

24.12.4.7 PreferredDepartureTimeCumulativeDistribution_T TRADEMGEN::DemandCharacteristics::_preferredDepartureTimeCumulativeDistribution

Preferred departure time cumulative distribution.

Definition at line 113 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#).

24.12.4.8 stdair::WTP_T TRADEMGEN::DemandCharacteristics::_minWTP

Min Willingness-to-pay, used for the computation of the WTP of each request.

Definition at line 119 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandStream::generateWTP\(\)](#).

24.12.4.9 CumulativeDistribution_T TRADEMGEN::DemandCharacteristics::_frat5Pattern

FRAT5 pattern, used for the computation of WTP.

Definition at line 124 of file [DemandCharacteristics.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateWTP\(\)](#).

24.12.4.10 ValueOfTimeCumulativeDistribution_T TRADEMGEN::DemandCharacteristics::_valueOfTimeCumulativeDistribution

Value of time cumulative distribution.

Definition at line 129 of file [DemandCharacteristics.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandStream::generateValueOfTime\(\)](#).

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

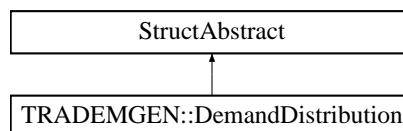
- [trademgen/basic/DemandCharacteristics.hpp](#)
- [trademgen/basic/DemandCharacteristics.cpp](#)

24.13 TRADEMGEN::DemandDistribution Struct Reference

Class modeling the distribution of a demand type.

```
#include <trademgen/basic/DemandDistribution.hpp>
```

Inheritance diagram for TRADEMGEN::DemandDistribution:



Public Member Functions

- [DemandDistribution](#) (const stdair::NbOfRequests_T &iMean, const stdair::StdDevValue_T &iStdDev)
- [DemandDistribution](#) ()
- [DemandDistribution](#) (const [DemandDistribution](#) &)
- [~DemandDistribution](#) ()
- void [fromStream](#) (std::istream &ioln)
- const std::string [describe](#) () const
- std::string [display](#) () const

Public Attributes

- stdair::NbOfRequests_T [_meanNumberOfRequests](#)
- stdair::StdDevValue_T [_stdDevNumberOfRequests](#)

24.13.1 Detailed Description

Class modeling the distribution of a demand type.

Definition at line 20 of file [DemandDistribution.hpp](#).

24.13.2 Constructor & Destructor Documentation

24.13.2.1 TRADEMGENT::DemandDistribution::DemandDistribution (const stdair::NbOfRequests_T &iMean, const stdair::StdDevValue_T &iStdDev)

Constructor.

Definition at line 15 of file [DemandDistribution.cpp](#).

24.13.2.2 TRADEMGEN::DemandDistribution::DemandDistribution ()

Default constructor.

Definition at line 22 of file [DemandDistribution.cpp](#).

24.13.2.3 TRADEMGEN::DemandDistribution::DemandDistribution (const DemandDistribution & iDemandDistribution)

Copy constructor.

Definition at line 31 of file [DemandDistribution.cpp](#).

24.13.2.4 TRADEMGEN::DemandDistribution::~~DemandDistribution ()

Destructor.

Definition at line 26 of file [DemandDistribution.cpp](#).

24.13.3 Member Function Documentation**24.13.3.1 void TRADEMGEN::DemandDistribution::fromStream (std::istream & ioln)**

Read a Business Object from an input stream.

Parameters

<i>istream&</i>	the input stream.
---------------------	-------------------

Definition at line 37 of file [DemandDistribution.cpp](#).

24.13.3.2 const std::string TRADEMGEN::DemandDistribution::describe () const

Display of the structure.

Definition at line 41 of file [DemandDistribution.cpp](#).

References [_meanNumberOfRequests](#), and [_stdDevNumberOfRequests](#).

Referenced by [display\(\)](#), and [TRADEMGEN::DemandStream::display\(\)](#).

24.13.3.3 std::string TRADEMGEN::DemandDistribution::display () const

Display demand distribution.

Definition at line 49 of file [DemandDistribution.cpp](#).

References [describe\(\)](#).

24.13.4 Member Data Documentation**24.13.4.1 stdair::NbOfRequests_T TRADEMGEN::DemandDistribution::_meanNumberOfRequests**

Mean number of requests.

Definition at line 67 of file [DemandDistribution.hpp](#).

Referenced by [describe\(\)](#), [TRADEMGEN::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#), and [TRADEMGEN::DemandStream::getMeanNumberOfRequests\(\)](#).

24.13.4.2 stdair::StdDevValue_T TRADEMGEN::DemandDistribution::_stdDevNumberOfRequests

Standard deviation of number of requests.

Definition at line 72 of file [DemandDistribution.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMGENT::DemandStream::getStdDevNumberOfRequests\(\)](#).

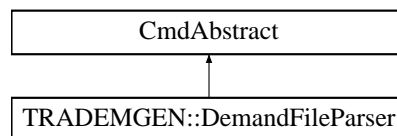
The documentation for this struct was generated from the following files:

- trademgen/basic/[DemandDistribution.hpp](#)
- trademgen/basic/[DemandDistribution.cpp](#)

24.14 TRADEMGENT::DemandFileParser Class Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGENT::DemandFileParser:



Public Member Functions

- [DemandFileParser](#) (stdair::EventQueue &, stdair::RandomGeneration &, const [POSProbabilityMass_T](#) &, const stdair::Filename_T & iDemandInputFilename)
- bool [generateDemand](#) ()

24.14.1 Detailed Description

Class wrapping the initialisation and entry point of the parser.

The seemingly redundancy is used to force the instantiation of the actual parser, which is a templatised Boost Spirit grammar. Hence, the actual parser is instantiated within that class object code.

Definition at line 393 of file [DemandParserHelper.hpp](#).

24.14.2 Constructor & Destructor Documentation

24.14.2.1 TRADEMGENT::DemandFileParser::DemandFileParser (stdair::EventQueue &, stdair::RandomGeneration &, const [POSProbabilityMass_T](#) &, const stdair::Filename_T & iDemandInputFilename)

Constructor.

Definition at line 791 of file [DemandParserHelper.cpp](#).

24.14.3 Member Function Documentation

24.14.3.1 bool TRADEMGENT::DemandFileParser::generateDemand ()

Parse the demand input file.

Definition at line 833 of file [DemandParserHelper.cpp](#).

Referenced by [TRADEMGENT::DemandParser::generateDemand\(\)](#).

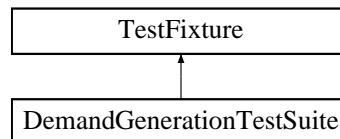
The documentation for this class was generated from the following files:

- trademgen/command/[DemandParserHelper.hpp](#)
- trademgen/command/[DemandParserHelper.cpp](#)

24.15 DemandGenerationTestSuite Class Reference

```
#include <test/trademgen/DemandGenerationTestSuite.hpp>
```

Inheritance diagram for DemandGenerationTestSuite:



Public Member Functions

- void [simpleEventGeneration](#) ()
- [DemandGenerationTestSuite](#) ()

Protected Attributes

- std::stringstream [_describeKey](#)

24.15.1 Detailed Description

Definition at line 6 of file [DemandGenerationTestSuite.hpp](#).

24.15.2 Constructor & Destructor Documentation

24.15.2.1 DemandGenerationTestSuite::DemandGenerationTestSuite ()

Test some error detection functionalities. Constructor.

24.15.3 Member Function Documentation

24.15.3.1 void DemandGenerationTestSuite::simpleEventGeneration ()

Test a simple event generation functionality.

24.15.4 Member Data Documentation

24.15.4.1 std::stringstream DemandGenerationTestSuite::_describeKey [protected]

Definition at line 27 of file [DemandGenerationTestSuite.hpp](#).

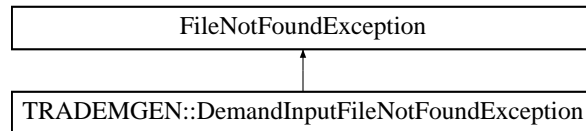
The documentation for this class was generated from the following file:

- test/trademgen/[DemandGenerationTestSuite.hpp](#)

24.16 TRADEMGEN::DemandInputFileNotFoundException Class Reference

```
#include <trademgen/TRADEMGEN_Exceptions.hpp>
```

Inheritance diagram for TRADEMGEN::DemandInputFileNotFoundException:



Public Member Functions

- [DemandInputFileNotFoundException](#) (const std::string &iWhat)

24.16.1 Detailed Description

Exception when no demand input file can be found

Definition at line 30 of file [TRADEMGEN_Exceptions.hpp](#).

24.16.2 Constructor & Destructor Documentation

24.16.2.1 TRADEMGEN::DemandInputFileNotFoundException::DemandInputFileNotFoundException (const std::string & iWhat) [inline]

Constructor.

Definition at line 36 of file [TRADEMGEN_Exceptions.hpp](#).

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

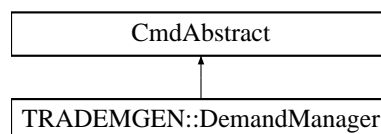
- trademgen/[TRADEMGEN_Exceptions.hpp](#)

24.17 TRADEMGEN::DemandManager Class Reference

Utility class for Demand and [DemandStream](#) objects.

```
#include <trademgen/command/DemandManager.hpp>
```

Inheritance diagram for TRADEMGEN::DemandManager:



Friends

- struct [DemandParserHelper::doEndDemand](#)
- class [TRADEMGEN_Service](#)

24.17.1 Detailed Description

Utility class for Demand and [DemandStream](#) objects.

Definition at line 38 of file [DemandManager.hpp](#).

24.17.2 Friends And Related Function Documentation

24.17.2.1 friend struct DemandParserHelper::doEndDemand [friend]

Definition at line 39 of file [DemandManager.hpp](#).

24.17.2.2 friend class TRADEMGEN_Service [friend]

Definition at line 40 of file [DemandManager.hpp](#).

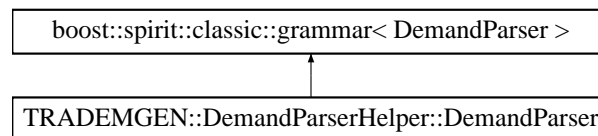
The documentation for this class was generated from the following files:

- [trademgen/command/DemandManager.hpp](#)
- [trademgen/command/DemandManager.cpp](#)

24.18 TRADEMGEN::DemandParserHelper::DemandParser Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::DemandParser:



Classes

- struct [definition](#)

Public Member Functions

- [DemandParser](#) (stdair::EventQueue &, stdair::RandomGeneration &, const [POSProbabilityMass_T](#) &, [DemandStruct](#) &)

Public Attributes

- stdair::EventQueue & [_eventQueue](#)
- stdair::RandomGeneration & [_uniformGenerator](#)
- const [POSProbabilityMass_T](#) & [_posProbabilityMass](#)
- [DemandStruct](#) & [_demand](#)

24.18.1 Detailed Description

PrefDepDate; Origin; Destination; PassengerType; Mean; StdDev; PosDist; ChannelDist; TripTypeDist; StayDurationDist; FrequentFlyerDist; PrefDepTimeDist; min WTP; (PrefArrivalDate; PrefArrivalTime;) TimeValueDist; ValueOfTimeDist; ArrivalPatternDist; 2010-02-08; SIN; BKK; L; 10.0; 1.0; SIN:0.7, BKK:0.2, row:0.1; DF:0.1, DN:0.3, IF:0.4, IN:0.2; RO:0.6, RI:0.2, OW:0.2; 0:0.1, 1:0.1, 2:0.15, 3:0.15, 4:0.15, 5:0.35; P:0.01, G:0.05, S:0.15, M:0.3, N:0.49; 06:0, 07:0.1, 09:0.3, 17:0.4, 19:0.8, 20:0.95, 22:1; 100:0, 500:0.8, 2000:1; 15:0, 60:1; 330:0, 40:0.2, 20:0.6, 1:1;

Fixed: Preferred departure date (yyyy-mm-dd) Origin (3-char airport code) Destination (3-char airport code) PassengerType (1-char, e.g., 'L' for Leisure, 'B' for Business) Observable: Mean StdDev Distribution with Probability Masses: POS Channel (D=direct, I=indirect, N=online, F=offline) Trip type(RO=outbound of round-trip, RI=inbound of round-trip, OW=one way) Stay duration (number of days) Frequent flyer (P=Platinum, G=Gold, S=Silver,

M=Member, N=None) Continuous cumulative distribution: Preferred departure time (hh:mm:ss) Preferred arrival date (equal to preferred departure date) Preferred arrival time (equal to preferred departure time) Value of time Arrival pattern (DTD as a positive value) The main fields are separated by ';' Probability mass distributions are defined by comma-separated 'value:probability' pairs Continuous cumulative distribution are defined by comma-separated 'value:probability' pairs, sorted in increasing order of values. The meaning of probability is $P(\text{random variable} \leq \text{value}) = \text{probability}$.

Grammar: Demand ::= PrefDepDate ';' Origin ';' Destination ';' PassengerType ';' DemandParams ';' PosDist ';' ChannelDist ';' TripDist ';' StayDist ';' FfDist ';' PrefDepTimeDist ';' minWTP ';' TimeValueDist ';' DtdDist EndOfDemand PrefDepDate ::= date PassengerType ::= 'L' | 'B' | 'F' DemandParams ::= DemandMean ';' DemandStdDev PosDist ::= PosPair (';' PosPair)* PosPair ::= PosCode ':' PosShare PosCode ::= AirportCode | "row" PosShare ::= real ChannelDist ::= ChannelPair (';' ChannelPair)* ChannelPair ::= Channel_Code ':' ChannelShare ChannelCode ::= "DF" | "DN" | "IF" | "IN" ChannelShare ::= real TripDist ::= TripPair (';' TripPair)* TripPair ::= TripCode ':' TripShare TripCode ::= "RO" | "RI" | "OW" TripShare ::= real StayDist ::= StayPair (';' StayPair)* StayPair ::= [0;3]-digit-integer ':' stay_share StayShare ::= real FfDist ::= FF_Pair (';' FF_Pair)* FFPair ::= FFCode ':' FFShare FFCode ::= 'P' | 'G' | 'S' | 'M' | 'N' FFShare ::= real PrefDepTimeDist ::= PrefDepTimePair (';' PrefDepTimePair)* PrefDepTimePair ::= time ':' PrefDepTimeShare PrefDepTimeShare ::= real minWTP ::= real TimeValueDist ::= TimeValuePair (';' TimeValuePair)* TimeValuePair ::= [0;2]-digit-integer ':' TimeValueShare TimeValueShare ::= real DTDDist ::= DTDPair (';' DTDPair)* DTDPair ::= real ':' DTDSHare DTDSHare ::= real EndOfDemand ::= ';' Grammar for the demand parser.

Definition at line 345 of file [DemandParserHelper.hpp](#).

24.18.2 Constructor & Destructor Documentation

24.18.2.1 TRADEMGEN::DemandParserHelper::DemandParser::DemandParser (stdair::EventQueue & ioEventQueue, stdair::RandomGeneration & ioSharedGenerator, const POSProbabilityMass_T & iPosProbMass, DemandStruct & ioDemand)

Definition at line 521 of file [DemandParserHelper.cpp](#).

24.18.3 Member Data Documentation

24.18.3.1 stdair::EventQueue& TRADEMGEN::DemandParserHelper::DemandParser::_eventQueue

Definition at line 374 of file [DemandParserHelper.hpp](#).

24.18.3.2 stdair::RandomGeneration& TRADEMGEN::DemandParserHelper::DemandParser::_uniformGenerator

Definition at line 375 of file [DemandParserHelper.hpp](#).

24.18.3.3 const POSProbabilityMass_T& TRADEMGEN::DemandParserHelper::DemandParser::_pos-ProbabilityMass

Definition at line 376 of file [DemandParserHelper.hpp](#).

24.18.3.4 DemandStruct& TRADEMGEN::DemandParserHelper::DemandParser::_demand

Definition at line 377 of file [DemandParserHelper.hpp](#).

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

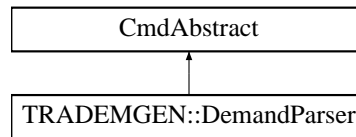
- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.19 TRADEMGEN::DemandParser Class Reference

Class wrapping the parser entry point.

```
#include <trademgen/command/DemandParser.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParser:



Static Public Member Functions

- static void [generateDemand](#) (const stdair::Filename_T &, stdair::EventQueue &, stdair::RandomGeneration &, const [POSProbabilityMass_T](#) &)

24.19.1 Detailed Description

Class wrapping the parser entry point.

Definition at line 26 of file [DemandParser.hpp](#).

24.19.2 Member Function Documentation

24.19.2.1 void **TRADEMGENT::DemandParser::generateDemand** (const stdair::Filename_T & *iFilename*, stdair::EventQueue & *ioEventQueue*, stdair::RandomGeneration & *ioSharedGenerator*, const **POSProbabilityMass_T** & *iDefaultPOSProbabilityMass*) [static]

Parse the CSV file describing travel demand, for instance for generating simulated booking request in a simulator.

The state of the random generator, given as parameter, evolves each time a demand request is generated.

Parameters

<i>const</i>	stdair::Filename_T& The file-name of the CSV-formatted demand input file.
<i>stdair::Event-Queue&</i>	Event queue.
<i>stdair::Random-Generation&</i>	Random generator.

Definition at line 18 of file [DemandParser.cpp](#).

References [TRADEMGENT::DemandFileParser::generateDemand\(\)](#).

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

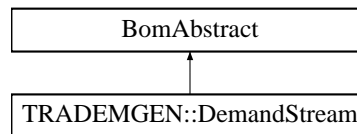
- trademgen/command/[DemandParser.hpp](#)
- trademgen/command/[DemandParser.cpp](#)

24.20 TRADEMGEN::DemandStream Class Reference

Class modeling a demand stream.

```
#include <trademgen/bom/DemandStream.hpp>
```

Inheritance diagram for TRADEMGEN::DemandStream:



Public Types

- typedef [DemandStreamKey](#) Key_T

Public Member Functions

- const [Key_T](#) & [getKey](#) () const
- [BomAbstract](#) *const [getParent](#) () const
- const stdair::AirportCode_T & [getOrigin](#) () const
- const stdair::AirportCode_T & [getDestination](#) () const
- const stdair::Date_T & [getPreferredDepartureDate](#) () const
- const stdair::CabinCode_T & [getPreferredCabin](#) () const
- const stdair::HolderMap_T & [getHolderMap](#) () const
- const [DemandCharacteristics](#) & [getDemandCharacteristics](#) () const
- const [DemandDistribution](#) & [getDemandDistribution](#) () const
- const stdair::NbOfRequests_T & [getTotalNumberOfRequestsToBeGenerated](#) () const
- const stdair::NbOfRequests_T & [getMeanNumberOfRequests](#) () const
- const stdair::StdDevValue_T & [getStdDevNumberOfRequests](#) () const
- const stdair::Count_T & [getNumberOfRequestsGeneratedSoFar](#) () const
- const [POSProbabilityMass_T](#) & [getPOSProbabilityMass](#) () const
- void [setNumberOfRequestsGeneratedSoFar](#) (const stdair::Count_T &iCount)
- void [setDemandDistribution](#) (const [DemandDistribution](#) &iDemandDistribution)
- void [setDemandCharacteristics](#) (const [ArrivalPatternCumulativeDistribution_T](#) &iArrivalPattern, const [POSProbabilityMassFunction_T](#) &iPOSProbMass, const [ChannelProbabilityMassFunction_T](#) &iChannelProbMass, const [TripTypeProbabilityMassFunction_T](#) &iTripTypeProbMass, const [StayDurationProbabilityMassFunction_T](#) &iStayDurationProbMass, const [FrequentFlyerProbabilityMassFunction_T](#) &iFrequentFlyerProbMass, const [PreferredDepartureTimeContinuousDistribution_T](#) &iPreferredDepartureTimeContinuousDistribution, const stdair::WTP_T &iMinWTP, const [ValueOfTimeContinuousDistribution_T](#) &iValueOfTimeContinuousDistribution)
- void [setTotalNumberOfRequestsToBeGenerated](#) (const stdair::NbOfRequests_T &iNbOfRequests)
- void [setRequestDateTimeRandomGeneratorSeed](#) (const stdair::RandomSeed_T &iSeed)
- void [setDemandCharacteristicsRandomGeneratorSeed](#) (const stdair::RandomSeed_T &iSeed)
- void [setPOSProbabilityMass](#) (const [POSProbabilityMass_T](#) &iProbMass)
- void [setAll](#) (const [ArrivalPatternCumulativeDistribution_T](#) &, const [POSProbabilityMassFunction_T](#) &, const [ChannelProbabilityMassFunction_T](#) &, const [TripTypeProbabilityMassFunction_T](#) &, const [StayDurationProbabilityMassFunction_T](#) &, const [FrequentFlyerProbabilityMassFunction_T](#) &, const [PreferredDepartureTimeContinuousDistribution_T](#) &, const stdair::WTP_T &, const [ValueOfTimeContinuousDistribution_T](#) &, const [DemandDistribution](#) &, stdair::BaseGenerator_T &ioSharedGenerator, const stdair::RandomSeed_T &iRequestDateTimeSeed, const stdair::RandomSeed_T &iDemandCharacteristicsSeed, const [POSProbabilityMass_T](#) &)
- void [setBoolFirstDateTimeRequest](#) (const bool &iFirstDateTimeRequest)
- void [incrementGeneratedRequestsCounter](#) ()
- const bool [stillHavingRequestsToBeGenerated](#) (const stdair::DemandGenerationMethod &iDemandGenerationMethod) const
- const stdair::DateTime_T [generateTimeOfRequestPoissonProcess](#) ()
- const stdair::DateTime_T [generateTimeOfRequestStatisticsOrder](#) ()
- const stdair::AirportCode_T [generatePOS](#) ()
- const stdair::ChannelLabel_T [generateChannel](#) ()

- const stdair::TripType_T [generateTripType](#) ()
- const stdair::DayDuration_T [generateStayDuration](#) ()
- const stdair::FrequentFlyer_T [generateFrequentFlyer](#) ()
- const stdair::Duration_T [generatePreferredDepartureTime](#) ()
- const stdair::WTP_T [generateWTP](#) (stdair::RandomGeneration &, const stdair::Date_T &, const stdair::Date-Time_T &, const stdair::DayDuration_T &)
- const stdair::PriceValue_T [generateValueOfTime](#) ()
- stdair::BookingRequestPtr_T [generateNextRequest](#) (stdair::RandomGeneration &, const stdair::Demand-GenerationMethod &)
- void [reset](#) (stdair::BaseGenerator_T &ioSharedGenerator)
- void [toStream](#) (std::ostream &ioOut) const
- void [fromStream](#) (std::istream &ioIn)
- std::string [toString](#) () const
- const std::string [describeKey](#) () const
- std::string [display](#) () const
- const stdair::Duration_T [convertFloatIntoDuration](#) (const stdair::FloatDuration_T)

Protected Member Functions

- [DemandStream](#) (const [Key_T](#) &)
- virtual [~DemandStream](#) ()

Protected Attributes

- [Key_T _key](#)
- [BomAbstract](#) * [_parent](#)
- stdair::HolderMap_T [_holderMap](#)
- [DemandCharacteristics](#) [_demandCharacteristics](#)
- [DemandDistribution](#) [_demandDistribution](#)
- stdair::NbOfRequests_T [_totalNumberOfRequestsToBeGenerated](#)
- [RandomGenerationContext](#) [_randomGenerationContext](#)
- stdair::RandomGeneration [_requestDateTimeRandomGenerator](#)
- stdair::RandomGeneration [_demandCharacteristicsRandomGenerator](#)
- [POSProbabilityMass_T](#) [_posProMass](#)

Friends

- class [stdair::FacBom](#)
- class [stdair::FacBomManager](#)

24.20.1 Detailed Description

Class modeling a demand stream.

Definition at line 30 of file [DemandStream.hpp](#).

24.20.2 Member Typedef Documentation

24.20.2.1 typedef DemandStreamKey TRADEMGEN::DemandStream::Key_T

Definition allowing to retrieve the associated BOM key type.

Definition at line 39 of file [DemandStream.hpp](#).

24.20.3 Constructor & Destructor Documentation

24.20.3.1 TRADEMGEN::DemandStream::DemandStream (const Key_T & iKey) [protected]

Main constructor.

Definition at line 62 of file [DemandStream.cpp](#).

24.20.3.2 TRADEMGEN::DemandStream::~DemandStream () [protected, virtual]

Destructor.

Definition at line 67 of file [DemandStream.cpp](#).

24.20.4 Member Function Documentation

24.20.4.1 const Key_T& TRADEMGEN::DemandStream::getKey () const [inline]

Get the key

Definition at line 45 of file [DemandStream.hpp](#).

References [_key](#).

24.20.4.2 BomAbstract* const TRADEMGEN::DemandStream::getParent () const [inline]

Get the parent object (EventQueue).

Definition at line 50 of file [DemandStream.hpp](#).

References [_parent](#).

24.20.4.3 const stdair::AirportCode_T& TRADEMGEN::DemandStream::getOrigin () const [inline]

Get the origin (part of the primary key).

Definition at line 55 of file [DemandStream.hpp](#).

References [_key](#), and [TRADEMGEN::DemandStreamKey::getOrigin\(\)](#).

24.20.4.4 const stdair::AirportCode_T& TRADEMGEN::DemandStream::getDestination () const [inline]

Get the destination (part of the primary key).

Definition at line 60 of file [DemandStream.hpp](#).

References [_key](#), and [TRADEMGEN::DemandStreamKey::getDestination\(\)](#).

24.20.4.5 const stdair::Date_T& TRADEMGEN::DemandStream::getPreferredDepartureDate () const [inline]

Get the preferred departure date (part of the primary key).

Definition at line 65 of file [DemandStream.hpp](#).

References [_key](#), and [TRADEMGEN::DemandStreamKey::getPreferredDepartureDate\(\)](#).

24.20.4.6 const stdair::CabinCode_T& TRADEMGEN::DemandStream::getPreferredCabin () const [inline]

Get the preferred cabin (part of the primary key).

Definition at line 70 of file [DemandStream.hpp](#).

References [_key](#), and [TRADEMGEN::DemandStreamKey::getPreferredCabin\(\)](#).

24.20.4.7 `const stdair::HolderMap_T& TRADEMGEN::DemandStream::getHolderMap () const [inline]`

Get the map of children holders.

Definition at line 75 of file [DemandStream.hpp](#).

References [_holderMap](#).

24.20.4.8 `const DemandCharacteristics& TRADEMGEN::DemandStream::getDemandCharacteristics () const [inline]`

Get the demand characteristics.

Definition at line 80 of file [DemandStream.hpp](#).

References [_demandCharacteristics](#).

24.20.4.9 `const DemandDistribution& TRADEMGEN::DemandStream::getDemandDistribution () const [inline]`

Get the demand distribution.

Definition at line 85 of file [DemandStream.hpp](#).

References [_demandDistribution](#).

24.20.4.10 `const stdair::NbOfRequests_T& TRADEMGEN::DemandStream::getTotalNumberOfRequestsToBeGenerated () const [inline]`

Get the total number of requests to be generated.

Definition at line 90 of file [DemandStream.hpp](#).

References [_totalNumberOfRequestsToBeGenerated](#).

24.20.4.11 `const stdair::NbOfRequests_T& TRADEMGEN::DemandStream::getMeanNumberOfRequests () const [inline]`

Get the mean (expected) number of requests.

Definition at line 95 of file [DemandStream.hpp](#).

References [_demandDistribution](#), and [TRADEMGENT::DemandDistribution::_meanNumberOfRequests](#).

24.20.4.12 `const stdair::StdDevValue_T& TRADEMGEN::DemandStream::getStdDevNumberOfRequests () const [inline]`

Get the standard deviation of number of requests.

Definition at line 100 of file [DemandStream.hpp](#).

References [_demandDistribution](#), and [TRADEMGENT::DemandDistribution::_stdDevNumberOfRequests](#).

24.20.4.13 `const stdair::Count_T& TRADEMGEN::DemandStream::getNumberOfRequestsGeneratedSoFar () const [inline]`

Get the number of requests generated so far.

Definition at line 105 of file [DemandStream.hpp](#).

References [_randomGenerationContext](#), and [TRADEMGENT::RandomGenerationContext::getNumberOfRequestsGeneratedSoFar\(\)](#).

24.20.4.14 `const POSProbabilityMass_T& TRADEMGEN::DemandStream::getPOSProbabilityMass () const [inline]`

Get the default POS probablity mass, used when "row" (rest of the world) is drawn.

Definition at line 113 of file [DemandStream.hpp](#).

References [_posProMass](#).

24.20.4.15 void TRADEMGEN::DemandStream::setNumberOfRequestsGeneratedSoFar (const stdair::Count_T & iCount) [inline]

Set the number of requests generated so far.

Definition at line 121 of file [DemandStream.hpp](#).

References [_randomGenerationContext](#), and [TRADEMGENT::RandomGenerationContext::setNumberOfRequestsGeneratedSoFar\(\)](#).

24.20.4.16 void TRADEMGEN::DemandStream::setDemandDistribution (const DemandDistribution & iDemandDistribution) [inline]

Set the demand distribution.

Definition at line 126 of file [DemandStream.hpp](#).

References [_demandDistribution](#).

Referenced by [setAll\(\)](#).

24.20.4.17 void TRADEMGEN::DemandStream::setDemandCharacteristics (const ArrivalPatternCumulativeDistribution_T & iArrivalPattern, const POSProbabilityMassFunction_T & iPOSProbMass, const ChannelProbabilityMassFunction_T & iChannelProbMass, const TripTypeProbabilityMassFunction_T & iTripTypeProbMass, const StayDurationProbabilityMassFunction_T & iStayDurationProbMass, const FrequentFlyerProbabilityMassFunction_T & iFrequentFlyerProbMass, const PreferredDepartureTimeContinuousDistribution_T & iPreferredDepartureTimeContinuousDistribution, const stdair::WTP_T & iMinWTP, const ValueOfTimeContinuousDistribution_T & iValueOfTimeContinuousDistribution) [inline]

Set the demand characteristics.

Definition at line 132 of file [DemandStream.hpp](#).

References [_demandCharacteristics](#).

Referenced by [setAll\(\)](#).

24.20.4.18 void TRADEMGEN::DemandStream::setTotalNumberOfRequestsToBeGenerated (const stdair::NbOfRequests_T & iNbOfRequests) [inline]

Set the total number of requests to be generated.

Definition at line 150 of file [DemandStream.hpp](#).

References [_totalNumberOfRequestsToBeGenerated](#).

Referenced by [setAll\(\)](#).

24.20.4.19 void TRADEMGEN::DemandStream::setRequestDateTimeRandomGeneratorSeed (const stdair::RandomSeed_T & iSeed) [inline]

Set the seed of the random generator for the request datetime.

Definition at line 155 of file [DemandStream.hpp](#).

References [_requestDateTimeRandomGenerator](#).

Referenced by [setAll\(\)](#).

24.20.4.20 void TRADEMGEN::DemandStream::setDemandCharacteristicsRandomGeneratorSeed (const stdair::RandomSeed_T & iSeed) [inline]

Set the seed of the random generator for the demand characteristics.

Definition at line 160 of file [DemandStream.hpp](#).

References [_demandCharacteristicsRandomGenerator](#).

Referenced by [setAll\(\)](#).

24.20.4.21 void TRADEMGEN::DemandStream::setPOSProbabilityMass (const POSProbabilityMass_T & iProbMass) [inline]

Set the default POS probability mass, used when "row" (rest of the world) is drawn.

Definition at line 168 of file [DemandStream.hpp](#).

References [_posProMass](#).

Referenced by [setAll\(\)](#).

24.20.4.22 void TRADEMGEN::DemandStream::setAll (const ArrivalPatternCumulativeDistribution_T & iArrivalPattern, const POSProbabilityMassFunction_T & iPOSProbMass, const ChannelProbabilityMassFunction_T & iChannelProbMass, const TripTypeProbabilityMassFunction_T & iTripTypeProbMass, const StayDurationProbabilityMassFunction_T & iStayDurationProbMass, const FrequentFlyerProbabilityMassFunction_T & iFrequentFlyerProbMass, const PreferredDepartureTimeContinuousDistribution_T & iPreferredDepartureTimeContinuousDistribution, const stdair::WTP_T & iMinWTP, const ValueOfTimeContinuousDistribution_T & iValueOfTimeContinuousDistribution, const DemandDistribution & iDemandDistribution, stdair::BaseGenerator_T & ioSharedGenerator, const stdair::RandomSeed_T & iRequestDateTimeSeed, const stdair::RandomSeed_T & iDemandCharacteristicsSeed, const POSProbabilityMass_T & iDefaultPOSProbabilityMass)

Initialisation.

Definition at line 79 of file [DemandStream.cpp](#).

References [setDemandCharacteristics\(\)](#), [setDemandCharacteristicsRandomGeneratorSeed\(\)](#), [setDemandDistribution\(\)](#), [setPOSProbabilityMass\(\)](#), [setRequestDateTimeRandomGeneratorSeed\(\)](#), and [setTotalNumberOfRequestsToBeGenerated\(\)](#).

24.20.4.23 void TRADEMGEN::DemandStream::setBoolFirstDateTimeRequest (const bool & iFirstDateTimeRequest) [inline]

Set the boolean describing if it is the first time we generate a request for a demand stream.

Definition at line 194 of file [DemandStream.hpp](#).

24.20.4.24 void TRADEMGEN::DemandStream::incrementGeneratedRequestsCounter () [inline]

Increment counter of requests generated so far

Definition at line 202 of file [DemandStream.hpp](#).

References [_randomGenerationContext](#), and [TRADEMGENT::RandomGenerationContext::incrementGeneratedRequestsCounter\(\)](#).

Referenced by [generateTimeOfRequestPoissonProcess\(\)](#), and [generateTimeOfRequestStatisticsOrder\(\)](#).

24.20.4.25 const bool TRADEMGEN::DemandStream::stillHavingRequestsToBeGenerated (const stdair::DemandGenerationMethod & iDemandGenerationMethod) const

Check whether enough requests have already been generated.

Definition at line 164 of file [DemandStream.cpp](#).

References [_randomGenerationContext](#), [_totalNumberOfRequestsToBeGenerated](#), and [TRADEMGENT::RandomGenerationContext::getNumberOfRequestsGeneratedSoFar\(\)](#).

24.20.4.26 `const std::pair<DateTime_T, TRADEMGEN::DemandStream::generateTimeOfRequestPoissonProcess ()`

Generate the time of the next request with poisson process.

Definition at line 189 of file [DemandStream.cpp](#).

References [TRADEMGENT::DemandCharacteristics::_arrivalPattern](#), [_demandCharacteristics](#), [_demandDistribution](#), [_key](#), [TRADEMGENT::DemandDistribution::_meanNumberOfRequests](#), [_requestDateTimeRandomGenerator](#), [convertFloatIntoDuration\(\)](#), [TRADEMGENT::DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN](#), [TRADEMGENT::ContinuousAttributeLite< T >::getDerivativeValue\(\)](#), [TRADEMGENT::DemandStreamKey::getPreferredDepartureDate\(\)](#), [TRADEMGENT::ContinuousAttributeLite< T >::getUpperBound\(\)](#), [TRADEMGENT::ContinuousAttributeLite< T >::getValue\(\)](#), and [incrementGeneratedRequestsCounter\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.27 `const std::pair<DateTime_T, TRADEMGEN::DemandStream::generateTimeOfRequestStatisticsOrder ()`

Generate the time of the next request with statistics order

Definition at line 291 of file [DemandStream.cpp](#).

References [TRADEMGENT::DemandCharacteristics::_arrivalPattern](#), [_demandCharacteristics](#), [_key](#), [_randomGenerationContext](#), [_requestDateTimeRandomGenerator](#), [_totalNumberOfRequestsToBeGenerated](#), [convertFloatIntoDuration\(\)](#), [TRADEMGENT::RandomGenerationContext::getCumulativeProbabilitySoFar\(\)](#), [TRADEMGENT::RandomGenerationContext::getNumberOfRequestsGeneratedSoFar\(\)](#), [TRADEMGENT::DemandStreamKey::getPreferredDepartureDate\(\)](#), [TRADEMGENT::ContinuousAttributeLite< T >::getValue\(\)](#), [incrementGeneratedRequestsCounter\(\)](#), and [TRADEMGENT::RandomGenerationContext::setCumulativeProbabilitySoFar\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.28 `const std::pair<AirportCode_T, TRADEMGEN::DemandStream::generatePOS ()`

Generate the POS.

Definition at line 422 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), and [TRADEMGENT::DemandCharacteristics::getPOSValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.29 `const std::pair<ChannelLabel_T, TRADEMGEN::DemandStream::generateChannel ()`

Generate the reservation channel.

Definition at line 432 of file [DemandStream.cpp](#).

References [TRADEMGENT::DemandCharacteristics::_channelProbabilityMass](#), [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), and [TRADEMGENT::CategoricalAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.30 `const std::pair<TripType_T, TRADEMGEN::DemandStream::generateTripType ()`

Generate the trip type.

Definition at line 441 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), [TRADEMGENT::DemandCharacteristics::_tripTypeProbabilityMass](#), and [TRADEMGENT::CategoricalAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.31 `const std::pair<DayDuration_T, TRADEMGEN::DemandStream::generateStayDuration ()`

Generate the stay duration.

Definition at line 450 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), [TRADEMGEN::DemandCharacteristics::_stayDurationProbabilityMass](#), and [TRADEMGEN::CategoricalAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.32 `const stdair::FrequentFlyer_T TRADEMGEN::DemandStream::generateFrequentFlyer ()`

Generate the frequent flyer type.

Definition at line 459 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), [TRADEMGEN::DemandCharacteristics::_frequentFlyerProbabilityMass](#), and [TRADEMGEN::CategoricalAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.33 `const stdair::Duration_T TRADEMGEN::DemandStream::generatePreferredDepartureTime ()`

Generate the preferred departure time.

Definition at line 468 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), and [_demandCharacteristicsRandomGenerator](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.34 `const stdair::WTP_T TRADEMGEN::DemandStream::generateWTP (stdair::RandomGeneration & ioGenerator, const stdair::Date_T & iDepartureDate, const stdair::DateTime_T & iDateTimeThisRequest, const stdair::DayDuration_T & iDurationOfStay)`

Generate the WTP.

Definition at line 482 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [TRADEMGEN::DemandCharacteristics::_frat5Pattern](#), [TRADEMGEN::DemandCharacteristics::_minWTP](#), and [TRADEMGEN::ContinuousAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.35 `const stdair::PriceValue_T TRADEMGEN::DemandStream::generateValueOfTime ()`

Generate the value of time.

Definition at line 503 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), [TRADEMGEN::DemandCharacteristics::_valueOfTimeCumulativeDistribution](#), and [TRADEMGEN::ContinuousAttributeLite< T >::getValue\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.36 `stdair::BookingRequestPtr_T TRADEMGEN::DemandStream::generateNextRequest (stdair::RandomGeneration & ioGenerator, const stdair::DemandGenerationMethod & iDemandGenerationMethod)`

Generate the next request.

Parameters

<i>stdair::RandomGeneration</i>	Random generator.
<i>const</i>	<code>stdair::DemandGenerationMethod::EN_DemandGenerationMethod</code> Method used to generate the date time of the next booking request: statistic order or poisson process.

Returns

stdair::BookingRequestPtr_T Next request to be simulate.

Definition at line 513 of file [DemandStream.cpp](#).

References [_key](#), [describeKey\(\)](#), [generateChannel\(\)](#), [generateFrequentFlyer\(\)](#), [generatePOS\(\)](#), [generatePreferredDepartureTime\(\)](#), [generateStayDuration\(\)](#), [generateTimeOfRequestPoissonProcess\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), [generateTripType\(\)](#), [generateValueOfTime\(\)](#), [generateWTP\(\)](#), [TRADEMGEN::DemandStreamKey::getDestination\(\)](#), [TRADEMGEN::DemandStreamKey::getOrigin\(\)](#), [TRADEMGEN::DemandStreamKey::getPreferredCabin\(\)](#), and [TRADEMGEN::DemandStreamKey::getPreferredDepartureDate\(\)](#).

24.20.4.37 void TRADEMGEN::DemandStream::reset (stdair::BaseGenerator_T & ioSharedGenerator)

Reset all the contexts of the demand stream.

Definition at line 589 of file [DemandStream.cpp](#).

References [_randomGenerationContext](#), and [TRADEMGEN::RandomGenerationContext::reset\(\)](#).

24.20.4.38 void TRADEMGEN::DemandStream::toStream (std::ostream & ioOut) const [inline]

Dump a Business Object into an output stream.

Parameters

<i>ostream&</i>	the output stream.
---------------------	--------------------

Definition at line 266 of file [DemandStream.hpp](#).

References [toString\(\)](#).

24.20.4.39 void TRADEMGEN::DemandStream::fromStream (std::istream & ioln) [inline]

Read a Business Object from an input stream.

Parameters

<i>istream&</i>	the input stream.
---------------------	-------------------

Definition at line 274 of file [DemandStream.hpp](#).

24.20.4.40 std::string TRADEMGEN::DemandStream::toString () const

Get the serialised version of the Business Object.

Definition at line 71 of file [DemandStream.cpp](#).

References [_key](#), and [TRADEMGEN::DemandStreamKey::toString\(\)](#).

Referenced by [toStream\(\)](#).

24.20.4.41 const std::string TRADEMGEN::DemandStream::describeKey () const [inline]

Get a string describing the key.

Definition at line 285 of file [DemandStream.hpp](#).

References [_key](#), and [TRADEMGEN::DemandStreamKey::toString\(\)](#).

Referenced by [generateNextRequest\(\)](#).

24.20.4.42 std::string TRADEMGEN::DemandStream::display () const

Dump recursively the content of the [DemandStream](#) object.

Definition at line 111 of file [DemandStream.cpp](#).

References [_demandCharacteristics](#), [_demandCharacteristicsRandomGenerator](#), [_demandDistribution](#), [_key](#), [_posProMass](#), [_randomGenerationContext](#), [_requestDateTimeRandomGenerator](#), [_totalNumberOfRequestsToBeGenerated](#), [TRADEMGENT::DemandCharacteristics::describe\(\)](#), [TRADEMGENT::DemandDistribution::describe\(\)](#), [TRADEMGENT::CategoricalAttributeLite< T >::displayProbabilityMass\(\)](#), and [TRADEMGENT::DemandStreamKey::toString\(\)](#).

Referenced by [TRADEMGENT::BomDisplay::csvDisplay\(\)](#).

24.20.4.43 `const std::duration<T> TRADEMGENT::DemandStream::convertFloatIntoDuration (const std::float_duration<T> iNumberOfDays)`

Definition at line 393 of file [DemandStream.cpp](#).

Referenced by [generateTimeOfRequestPoissonProcess\(\)](#), and [generateTimeOfRequestStatisticsOrder\(\)](#).

24.20.5 Friends And Related Function Documentation

24.20.5.1 `friend class std::facBom [friend]`

Definition at line 31 of file [DemandStream.hpp](#).

24.20.5.2 `friend class std::facBomManager [friend]`

Definition at line 32 of file [DemandStream.hpp](#).

24.20.6 Member Data Documentation

24.20.6.1 `Key_T TRADEMGENT::DemandStream::_key [protected]`

Primary key (string gathering the origin, destination, POS and date).

Definition at line 321 of file [DemandStream.hpp](#).

Referenced by [describeKey\(\)](#), [display\(\)](#), [generateNextRequest\(\)](#), [generateTimeOfRequestPoissonProcess\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), [getDestination\(\)](#), [getKey\(\)](#), [getOrigin\(\)](#), [getPreferredCabin\(\)](#), [getPreferredDepartureDate\(\)](#), and [toString\(\)](#).

24.20.6.2 `BomAbstract* TRADEMGENT::DemandStream::_parent [protected]`

Pointer on the parent class (EventQueue).

Definition at line 326 of file [DemandStream.hpp](#).

Referenced by [getParent\(\)](#).

24.20.6.3 `std::holder_map<T> TRADEMGENT::DemandStream::_holderMap [protected]`

Map holding the children (not used for now).

Definition at line 331 of file [DemandStream.hpp](#).

Referenced by [getHolderMap\(\)](#).

24.20.6.4 `DemandCharacteristics TRADEMGENT::DemandStream::_demandCharacteristics [protected]`

Demand characteristics.

Definition at line 336 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateChannel\(\)](#), [generateFrequentFlyer\(\)](#), [generatePOS\(\)](#), [generatePreferredDepartureTime\(\)](#), [generateStayDuration\(\)](#), [generateTimeOfRequestPoissonProcess\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), [generateTripType\(\)](#), [generateValueOfTime\(\)](#), [generateWTP\(\)](#), [getDemandCharacteristics\(\)](#), and [setDemandCharacteristics\(\)](#).

24.20.6.5 DemandDistribution TRADEMGEN::DemandStream::_demandDistribution [protected]

Demand distribution.

Definition at line 341 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateTimeOfRequestPoissonProcess\(\)](#), [getDemandDistribution\(\)](#), [getMeanNumberOfRequests\(\)](#), [getStdDevNumberOfRequests\(\)](#), and [setDemandDistribution\(\)](#).

24.20.6.6 stdair::NbOfRequests_T TRADEMGEN::DemandStream::_totalNumberOfRequestsToBeGenerated [protected]

Total number of requests to be generated.

Definition at line 346 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), [getTotalNumberOfRequestsToBeGenerated\(\)](#), [setTotalNumberOfRequestsToBeGenerated\(\)](#), and [stillHavingRequestsToBeGenerated\(\)](#).

24.20.6.7 RandomGenerationContext TRADEMGEN::DemandStream::_randomGenerationContext [protected]

Random generation context.

Definition at line 351 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), [getNumberOfRequestsGeneratedSoFar\(\)](#), [incrementGeneratedRequestsCounter\(\)](#), [reset\(\)](#), [setNumberOfRequestsGeneratedSoFar\(\)](#), and [stillHavingRequestsToBeGenerated\(\)](#).

24.20.6.8 stdair::RandomGeneration TRADEMGEN::DemandStream::_requestDateTimeRandomGenerator [protected]

Random generator for request date-time.

Definition at line 356 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateTimeOfRequestPoissonProcess\(\)](#), [generateTimeOfRequestStatisticsOrder\(\)](#), and [setRequestDateTimeRandomGeneratorSeed\(\)](#).

24.20.6.9 stdair::RandomGeneration TRADEMGEN::DemandStream::_demandCharacteristicsRandomGenerator [protected]

Random generator for demand characteristics.

Definition at line 361 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [generateChannel\(\)](#), [generateFrequentFlyer\(\)](#), [generatePOS\(\)](#), [generatePreferredDepartureTime\(\)](#), [generateStayDuration\(\)](#), [generateTripType\(\)](#), [generateValueOfTime\(\)](#), and [setDemandCharacteristicsRandomGeneratorSeed\(\)](#).

24.20.6.10 POSProbabilityMass_T TRADEMGEN::DemandStream::_posProMass [protected]

Default POS probability mass, used when "row" (rest of the world) is drawn.

Definition at line 367 of file [DemandStream.hpp](#).

Referenced by [display\(\)](#), [getPOSProbabilityMass\(\)](#), and [setPOSProbabilityMass\(\)](#).

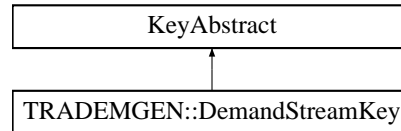
The documentation for this class was generated from the following files:

- [trademgen/bom/DemandStream.hpp](#)
- [trademgen/bom/DemandStream.cpp](#)

24.21 TRADEMGEN::DemandStreamKey Struct Reference

```
#include <trademgen/bom/DemandStreamKey.hpp>
```

Inheritance diagram for TRADEMGEN::DemandStreamKey:



Public Member Functions

- [DemandStreamKey](#) (const stdair::AirportCode_T &iOrigin, const stdair::AirportCode_T &iDestination, const stdair::Date_T &iPreferredDepartureDate, const stdair::CabinCode_T &iPreferredCabin)
- [DemandStreamKey](#) (const [DemandStreamKey](#) &)
- [~DemandStreamKey](#) ()
- const stdair::AirportCode_T & [getOrigin](#) () const
- const stdair::AirportCode_T & [getDestination](#) () const
- const stdair::Date_T & [getPreferredDepartureDate](#) () const
- const stdair::CabinCode_T & [getPreferredCabin](#) () const
- void [toStream](#) (std::ostream &ioOut) const
- void [fromStream](#) (std::istream &ioIn)
- const std::string [toString](#) () const

24.21.1 Detailed Description

Key of a given demand-stream, made of a pair of required airports/cities (origin and destination), a preferred departure date and a preferred cabin. Those attributes correspond to a the travel requirements of a simulated traveller.

Definition at line 20 of file [DemandStreamKey.hpp](#).

24.21.2 Constructor & Destructor Documentation

24.21.2.1 TRADEMGEN::DemandStreamKey::DemandStreamKey (const stdair::AirportCode_T &iOrigin, const stdair::AirportCode_T &iDestination, const stdair::Date_T &iPreferredDepartureDate, const stdair::CabinCode_T &iPreferredCabin)

Constructor.

Definition at line 25 of file [DemandStreamKey.cpp](#).

24.21.2.2 TRADEMGEN::DemandStreamKey::DemandStreamKey (const DemandStreamKey &iKey)

Default copy constructor.

Definition at line 35 of file [DemandStreamKey.cpp](#).

24.21.2.3 TRADEMGEN::DemandStreamKey::~~DemandStreamKey ()

Destructor.

Definition at line 42 of file [DemandStreamKey.cpp](#).

24.21.3 Member Function Documentation

24.21.3.1 const stdair::AirportCode_T& TRADEMGEN::DemandStreamKey::getOrigin () const [inline]

Get the origin.

Definition at line 43 of file [DemandStreamKey.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateNextRequest\(\)](#), and [TRADEMGENT::DemandStream::getOrigin\(\)](#).

24.21.3.2 const stdair::AirportCode_T& TRADEMGEN::DemandStreamKey::getDestination () const [inline]

Get the destination.

Definition at line 48 of file [DemandStreamKey.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateNextRequest\(\)](#), and [TRADEMGENT::DemandStream::getDestination\(\)](#).

24.21.3.3 const stdair::Date_T& TRADEMGEN::DemandStreamKey::getPreferredDepartureDate () const [inline]

Get the preferred departure date.

Definition at line 53 of file [DemandStreamKey.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateNextRequest\(\)](#), [TRADEMGENT::DemandStream::generateTimeOfRequestPoissonProcess\(\)](#), [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#), and [TRADEMGENT::DemandStream::getPreferredDepartureDate\(\)](#).

24.21.3.4 const stdair::CabinCode_T& TRADEMGEN::DemandStreamKey::getPreferredCabin () const [inline]

Get the preferred cabin.

Definition at line 58 of file [DemandStreamKey.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateNextRequest\(\)](#), and [TRADEMGENT::DemandStream::getPreferredCabin\(\)](#).

24.21.3.5 void TRADEMGEN::DemandStreamKey::toStream (std::ostream & ioOut) const

Dump a Business Object Key into an output stream.

Parameters

<i>ostream&</i>	the output stream.
---------------------	--------------------

Definition at line 46 of file [DemandStreamKey.cpp](#).

References [toString\(\)](#).

24.21.3.6 void TRADEMGEN::DemandStreamKey::fromStream (std::istream & ioIn)

Read a Business Object Key from an input stream.

Parameters

<i>istream&</i>	the input stream.
---------------------	-------------------

Definition at line 51 of file [DemandStreamKey.cpp](#).

24.21.3.7 `const std::string TRADEMGEN::DemandStreamKey::toString () const`

Get the serialised version of the Business Object Key.

That string is unique, at the level of a given Business Object, when among children of a given parent Business Object.

For instance, "H" and "K" allow to differentiate among two marketing classes for the same segment-stream.

Definition at line 55 of file [DemandStreamKey.cpp](#).

Referenced by [TRADEMGENT::DemandStream::describeKey\(\)](#), [TRADEMGENT::DemandStream::display\(\)](#), [toStream\(\)](#), and [TRADEMGENT::DemandStream::toString\(\)](#).

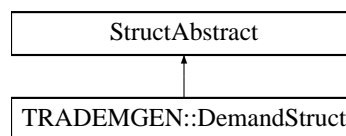
The documentation for this struct was generated from the following files:

- [trademgen/bom/DemandStreamKey.hpp](#)
- [trademgen/bom/DemandStreamKey.cpp](#)

24.22 TRADEMGEN::DemandStruct Struct Reference

```
#include <trademgen/bom/DemandStruct.hpp>
```

Inheritance diagram for TRADEMGEN::DemandStruct:



Public Member Functions

- `stdair::Date_T` [getDate](#) () const
- `stdair::Duration_T` [getTime](#) () const
- `const std::string` [describe](#) () const
- [DemandStruct](#) ()
- [~DemandStruct](#) ()

Public Attributes

- `stdair::DatePeriod_T` [_dateRange](#)
- `stdair::DoWStruct` [_dow](#)
- `stdair::AirportCode_T` [_origin](#)
- `stdair::AirportCode_T` [_destination](#)
- `stdair::CabinCode_T` [_prefCabin](#)
- `stdair::MeanValue_T` [_demandMean](#)
- `stdair::StdDevValue_T` [_demandStdDev](#)
- `POSProbabilityMassFunction_T` [_posProbDist](#)
- `ChannelProbabilityMassFunction_T` [_channelProbDist](#)
- `TripTypeProbabilityMassFunction_T` [_tripProbDist](#)
- `StayDurationProbabilityMassFunction_T` [_stayProbDist](#)
- `FrequentFlyerProbabilityMassFunction_T` [_ffProbDist](#)
- `PreferredDepartureTimeContinuousDistribution_T` [_prefDepTimeProbDist](#)
- `stdair::WTP_T` [_minWTP](#)
- `ValueOfTimeContinuousDistribution_T` [_timeValueProbDist](#)
- `ArrivalPatternCumulativeDistribution_T` [_dtdProbDist](#)

- stdair::Date_T _prefDepDateStart
- stdair::Date_T _prefDepDateEnd
- unsigned int _itYear
- unsigned int _itMonth
- unsigned int _itDay
- long _itHours
- long _itMinutes
- long _itSeconds
- stdair::AirportCode_T _itPosCode
- stdair::ChannelLabel_T _itChannelCode
- stdair::TripType_T _itTripCode
- stdair::DayDuration_T _itStayDuration
- stdair::FrequentFlyer_T _itFFCode
- stdair::Duration_T _itPrefDepTime
- stdair::PriceValue_T _itTimeValue
- stdair::DayDuration_T _itDTD

24.22.1 Detailed Description

Utility Structure for the parsing of Demand structures.

Definition at line 21 of file [DemandStruct.hpp](#).

24.22.2 Constructor & Destructor Documentation

24.22.2.1 TRADEMGEN::DemandStruct::DemandStruct ()

Default constructor.

Definition at line 18 of file [DemandStruct.cpp](#).

24.22.2.2 TRADEMGEN::DemandStruct::~~DemandStruct ()

Destructor

Definition at line 26 of file [DemandStruct.cpp](#).

24.22.3 Member Function Documentation

24.22.3.1 stdair::Date_T TRADEMGEN::DemandStruct::getDate () const

Get the date from the staging details.

Definition at line 30 of file [DemandStruct.cpp](#).

References [_itDay](#), [_itMonth](#), and [_itYear](#).

Referenced by [TRADEMG-EN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)\(\)](#), and [TRADEMG-EN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#).

24.22.3.2 stdair::Duration_T TRADEMGEN::DemandStruct::getTime () const

Get the time from the staging details.

Definition at line 35 of file [DemandStruct.cpp](#).

References [_itHours](#), [_itMinutes](#), and [_itSeconds](#).

Referenced by [TRADEMG-EN::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#).

24.22.3.3 `const std::string TRADEMGEN::DemandStruct::describe () const`

Give a description of the structure (for display purposes).

Definition at line 42 of file [DemandStruct.cpp](#).

References [_channelProbDist](#), [_dateRange](#), [_demandMean](#), [_demandStdDev](#), [_destination](#), [_dow](#), [_dtdProbDist](#), [_ffProbDist](#), [_minWTP](#), [_origin](#), [_posProbDist](#), [_prefCabin](#), [_prefDepTimeProbDist](#), [_stayProbDist](#), [_timeValueProbDist](#), and [_tripProbDist](#).

24.22.4 Member Data Documentation

24.22.4.1 `stdair::DatePeriod_T TRADEMGEN::DemandStruct::_dateRange`

Definition at line 51 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMG-EN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#).

24.22.4.2 `stdair::DoWStruct TRADEMGEN::DemandStruct::_dow`

Definition at line 52 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMG-EN::DemandParserHelper::storeDow::operator\(\)](#).

24.22.4.3 `stdair::AirportCode_T TRADEMGEN::DemandStruct::_origin`

Definition at line 53 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMG-EN::DemandParserHelper::storeOrigin::operator\(\)](#).

24.22.4.4 `stdair::AirportCode_T TRADEMGEN::DemandStruct::_destination`

Definition at line 54 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMG-EN::DemandParserHelper::storeDestination::operator\(\)](#).

24.22.4.5 `stdair::CabinCode_T TRADEMGEN::DemandStruct::_prefCabin`

Definition at line 55 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMG-EN::DemandParserHelper::storePrefCabin::operator\(\)](#).

24.22.4.6 `stdair::MeanValue_T TRADEMGEN::DemandStruct::_demandMean`

Definition at line 56 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMG-EN::DemandParserHelper::storeDemandMean::operator\(\)](#).

24.22.4.7 `stdair::StdDevValue_T TRADEMGEN::DemandStruct::_demandStdDev`

Definition at line 57 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMG-EN::DemandParserHelper::storeDemandStdDev::operator\(\)](#).

24.22.4.8 `POSProbabilityMassFunction_T TRADEMGEN::DemandStruct::_posProbDist`

Definition at line 58 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMG-EN::DemandParserHelper::storePosProbMass::operator\(\)](#), and [TRADEMG-EN::DemandParserHelper::doEndDemand::operator\(\)](#).

24.22.4.9 `ChannelProbabilityMassFunction_T TRADEMGEN::DemandStruct::_channelProbDist`

Definition at line 59 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMG-EN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), and [TRADEMG-EN::DemandParserHelper::doEndDemand::operator\(\)](#).

[EMGEN::DemandParserHelper::doEndDemand::operator\(\)](#).

24.22.4.10 `TripTypeProbabilityMassFunction_T` `TRADEMG-EN::DemandStruct::_tripProbDist`

Definition at line 60 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMG-EN::DemandParserHelper::storeTripProbMass::operator\(\)](#), and [TRADEMG-EN::DemandParserHelper::doEndDemand::operator\(\)](#).

24.22.4.11 `StayDurationProbabilityMassFunction_T` `TRADEMG-EN::DemandStruct::_stayProbDist`

Definition at line 61 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMG-EN::DemandParserHelper::storeStayProbMass::operator\(\)](#), and [TRADEMG-EN::DemandParserHelper::doEndDemand::operator\(\)](#).

24.22.4.12 `FrequentFlyerProbabilityMassFunction_T` `TRADEMG-EN::DemandStruct::_ffProbDist`

Definition at line 62 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMG-EN::DemandParserHelper::storeFFProbMass::operator\(\)](#), and [TRADEMG-EN::DemandParserHelper::doEndDemand::operator\(\)](#).

24.22.4.13 `PreferredDepartureTimeContinuousDistribution_T` `TRADEMG-EN::DemandStruct::_prefDepTimeProbDist`

Definition at line 63 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMG-EN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), and [TRADEMG-EN::DemandParserHelper::doEndDemand::operator\(\)](#).

24.22.4.14 `stdair::WTP_T` `TRADEMG-EN::DemandStruct::_minWTP`

Definition at line 64 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), and [TRADEMG-EN::DemandParserHelper::storeWTP::operator\(\)](#).

24.22.4.15 `ValueOfTimeContinuousDistribution_T` `TRADEMG-EN::DemandStruct::_timeValueProbDist`

Definition at line 65 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMG-EN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), and [TRADEMG-EN::DemandParserHelper::doEndDemand::operator\(\)](#).

24.22.4.16 `ArrivalPatternCumulativeDistribution_T` `TRADEMG-EN::DemandStruct::_dtdProbDist`

Definition at line 66 of file [DemandStruct.hpp](#).

Referenced by [describe\(\)](#), [TRADEMG-EN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMG-EN::DemandParserHelper::doEndDemand::operator\(\)](#).

24.22.4.17 `stdair::Date_T` `TRADEMG-EN::DemandStruct::_prefDepDateStart`

Staging Date.

Definition at line 71 of file [DemandStruct.hpp](#).

Referenced by [TRADEMG-EN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), and [TRADEMG-EN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#).

24.22.4.18 `stdair::Date_T` `TRADEMG-EN::DemandStruct::_prefDepDateEnd`

Definition at line 72 of file [DemandStruct.hpp](#).

Referenced by [TRADEMG-EN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#).

24.22.4.19 unsigned int TRADEMGEN::DemandStruct::_itYear

Definition at line 73 of file [DemandStruct.hpp](#).

Referenced by [getDate\(\)](#).

24.22.4.20 unsigned int TRADEMGEN::DemandStruct::_itMonth

Definition at line 74 of file [DemandStruct.hpp](#).

Referenced by [getDate\(\)](#).

24.22.4.21 unsigned int TRADEMGEN::DemandStruct::_itDay

Definition at line 75 of file [DemandStruct.hpp](#).

Referenced by [getDate\(\)](#).

24.22.4.22 long TRADEMGEN::DemandStruct::_itHours

Staging Time.

Definition at line 78 of file [DemandStruct.hpp](#).

Referenced by [getTime\(\)](#).

24.22.4.23 long TRADEMGEN::DemandStruct::_itMinutes

Definition at line 79 of file [DemandStruct.hpp](#).

Referenced by [getTime\(\)](#), and [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#).

24.22.4.24 long TRADEMGEN::DemandStruct::_itSeconds

Definition at line 80 of file [DemandStruct.hpp](#).

Referenced by [getTime\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#).

24.22.4.25 stdair::AirportCode_T TRADEMGEN::DemandStruct::_itPosCode

Staging Point-Of-Sale (POS) code.

Definition at line 83 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#).

24.22.4.26 stdair::ChannelLabel_T TRADEMGEN::DemandStruct::_itChannelCode

Staging channel type code.

Definition at line 86 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#).

24.22.4.27 stdair::TripType_T TRADEMGEN::DemandStruct::_itTripCode

Staging trip type code.

Definition at line 89 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#).

24.22.4.28 stdair::DayDuration_T TRADEMGEN::DemandStruct::_itStayDuration

Staging stay duration.

Definition at line 92 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#).

24.22.4.29 stdair::FrequentFlyer_T TRADEMGEN::DemandStruct::_itFFCode

Staging Frequent Flyer code.

Definition at line 95 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#).

24.22.4.30 stdair::Duration_T TRADEMGEN::DemandStruct::_itPrefDepTime

Staging preferred departure time.

Definition at line 98 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#).

24.22.4.31 stdair::PriceValue_T TRADEMGEN::DemandStruct::_itTimeValue

Staging time value.

Definition at line 101 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#).

24.22.4.32 stdair::DayDuration_T TRADEMGEN::DemandStruct::_itDTD

Staging DTD (Days-To-Departure).

Definition at line 104 of file [DemandStruct.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#).

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

- [trademgen/bom/DemandStruct.hpp](#)
- [trademgen/bom/DemandStruct.cpp](#)

24.23 TRADEMGEN::DictionaryManager Class Reference

Class wrapper of dictionary business methods.

```
#include <trademgen/basic/DictionaryManager.hpp>
```

Static Public Member Functions

- static const stdair::Probability_T [keyToValue](#) (const [DictionaryKey_T](#))
- static const [DictionaryKey_T](#) [valueToKey](#) (const stdair::Probability_T)

24.23.1 Detailed Description

Class wrapper of dictionary business methods.

Definition at line 21 of file [DictionaryManager.hpp](#).

24.23.2 Member Function Documentation

24.23.2.1 `const stdair::Probability_T TRADEMGEN::DictionaryManager::keyToValue (const DictionaryKey_T iKey) [static]`

Convert from key to value.

Definition at line 10 of file [DictionaryManager.cpp](#).

Referenced by [TRADEMGENT::ContinuousAttribute< T >::displayCumulativeDistribution\(\)](#), [TRADEMGENT::ContinuousAttributeLite< stdair::FloatDuration_T >::displayCumulativeDistribution\(\)](#), [TRADEMGENT::ContinuousAttribute< T >::displayInverseCumulativeDistribution\(\)](#), [TRADEMGENT::CategoricalAttributeLite< stdair::TripType_T >::displayProbabilityMass\(\)](#), [TRADEMGENT::ContinuousAttributeLite< stdair::FloatDuration_T >::getDerivativeValue\(\)](#), [TRADEMGENT::ContinuousAttributeLite< stdair::FloatDuration_T >::getValue\(\)](#), and [TRADEMGENT::ContinuousAttribute< T >::getValue\(\)](#).

24.23.2.2 `const DictionaryKey_T TRADEMGEN::DictionaryManager::valueToKey (const stdair::Probability_T iValue) [static]`

Convert from value to key.

Definition at line 17 of file [DictionaryManager.cpp](#).

Referenced by [TRADEMGENT::ContinuousAttributeLite< stdair::FloatDuration_T >::getValue\(\)](#), [TRADEMGENT::CategoricalAttributeLite< stdair::TripType_T >::getValue\(\)](#), and [TRADEMGENT::ContinuousAttribute< T >::getValue\(\)](#).

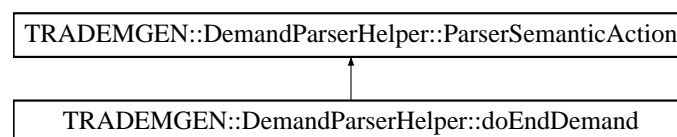
The documentation for this class was generated from the following files:

- [trademgen/basic/DictionaryManager.hpp](#)
- [trademgen/basic/DictionaryManager.cpp](#)

24.24 TRADEMGEN::DemandParserHelper::doEndDemand Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::doEndDemand:



Public Member Functions

- `doEndDemand` (stdair::EventQueue &, stdair::RandomGeneration &, const [POSPProbabilityMass_T](#) &, [DemandStruct](#) &)
- `void operator() (iterator_t iStr, iterator_t iStrEnd) const`

Public Attributes

- `stdair::EventQueue & _eventQueue`
- `stdair::RandomGeneration & _uniformGenerator`
- `const POSPProbabilityMass_T & _posProbabilityMass`
- `DemandStruct & _demand`

24.24.1 Detailed Description

Mark the end of the demand parsing.

Definition at line 242 of file [DemandParserHelper.hpp](#).

24.24.2 Constructor & Destructor Documentation

24.24.2.1 TRADEMGEN::DemandParserHelper::doEndDemand::doEndDemand (*stdair::EventQueue & ioEventQueue*, *stdair::RandomGeneration & ioSharedGenerator*, *const POSProbabilityMass_T & iPOSProbMass*, *DemandStruct & ioDemand*)

Actor Constructor.

Definition at line 412 of file [DemandParserHelper.cpp](#).

24.24.3 Member Function Documentation

24.24.3.1 void TRADEMGEN::DemandParserHelper::doEndDemand::operator() (*iterator_t iStr*, *iterator_t iStrEnd*) const

Actor Function (functor).

Definition at line 423 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandStruct::_channelProbDist](#), [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_dtdProbDist](#), [_eventQueue](#), [TRADEMGEN::DemandStruct::_ffProbDist](#), [_posProbabilityMass](#), [TRADEMGEN::DemandStruct::_posProbDist](#), [TRADEMGEN::DemandStruct::_prefDepTimeProbDist](#), [TRADEMGEN::DemandStruct::_stayProbDist](#), [TRADEMGEN::DemandStruct::_timeValueProbDist](#), [TRADEMGEN::DemandStruct::_tripProbDist](#), and [_uniformGenerator](#).

24.24.4 Member Data Documentation

24.24.4.1 *stdair::EventQueue&* TRADEMGEN::DemandParserHelper::doEndDemand::_eventQueue

Actor Specific Context.

Definition at line 249 of file [DemandParserHelper.hpp](#).

Referenced by [operator\(\)\(\)](#).

24.24.4.2 *stdair::RandomGeneration&* TRADEMGEN::DemandParserHelper::doEndDemand::_uniformGenerator

Definition at line 250 of file [DemandParserHelper.hpp](#).

Referenced by [operator\(\)\(\)](#).

24.24.4.3 *const POSProbabilityMass_T&* TRADEMGEN::DemandParserHelper::doEndDemand::_pos-ProbabilityMass

Definition at line 251 of file [DemandParserHelper.hpp](#).

Referenced by [operator\(\)\(\)](#).

24.24.4.4 *DemandStruct&* TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#), [TRADEMGEN::DemandParserHelper::store-](#)

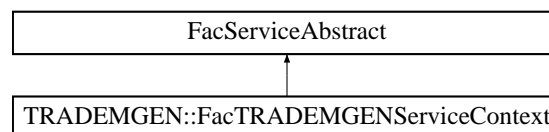
[Dow::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#)(), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#)(), and [operator\(\)](#).

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

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.25 FacServiceAbstract Class Reference

Inheritance diagram for FacServiceAbstract:



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

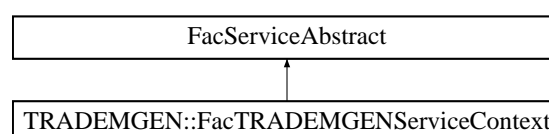
- [trademgen/factory/FacTRADEMGENServiceContext.hpp](#)

24.26 TRADEMGEN::FacTRADEMGENServiceContext Class Reference

Factory for creating the TraDemGen service context instance.

```
#include <trademgen/factory/FacTRADEMGENServiceContext.hpp>
```

Inheritance diagram for TRADEMGEN::FacTRADEMGENServiceContext:



Public Member Functions

- [~FacTRADEMGENServiceContext\(\)](#)
- [TRADEMGEN_ServiceContext & create](#) (const stdair::RandomSeed_T &)

Static Public Member Functions

- static [FacTRADEMGENSEerviceContext](#) & [instance](#) ()

Protected Member Functions

- [FacTRADEMGENSEerviceContext](#) ()

24.26.1 Detailed Description

Factory for creating the TraDemGen service context instance.

Definition at line 21 of file [FacTRADEMGENSEerviceContext.hpp](#).

24.26.2 Constructor & Destructor Documentation

24.26.2.1 TRADEMGEN::FacTRADEMGENSEerviceContext::~~FacTRADEMGENSEerviceContext ()

Destructor.

The Destruction put the `_instance` to NULL in order to be clean for the next [FacTRADEMGENSEerviceContext::instance\(\)](#).

Definition at line 17 of file [FacTRADEMGENSEerviceContext.cpp](#).

24.26.2.2 TRADEMGEN::FacTRADEMGENSEerviceContext::FacTRADEMGENSEerviceContext ()
[inline, protected]

Default Constructor.

This constructor is protected in order to ensure the singleton pattern.

Definition at line 57 of file [FacTRADEMGENSEerviceContext.hpp](#).

Referenced by [instance\(\)](#).

24.26.3 Member Function Documentation

24.26.3.1 FacTRADEMGENSEerviceContext & TRADEMGEN::FacTRADEMGENSEerviceContext::instance ()
[static]

Provide the unique instance.

The singleton is instantiated when first used.

Returns

[FacTRADEMGENSEerviceContext](#)&

Definition at line 22 of file [FacTRADEMGENSEerviceContext.cpp](#).

References [FacTRADEMGENSEerviceContext\(\)](#).

24.26.3.2 TRADEMGEN_ServiceContext & TRADEMGEN::FacTRADEMGENSEerviceContext::create (const
stdair::RandomSeed_T & *iRandomSeed*)

Create a new [TRADEMGEN_ServiceContext](#) object.

This new object is added to the list of instantiated objects.

Parameters

<i>const</i>	stdair::RandomSeed_T& Seed for the random generation.
--------------	---

Returns

[TRADEMGEN_ServiceContext](#)& The newly created object.

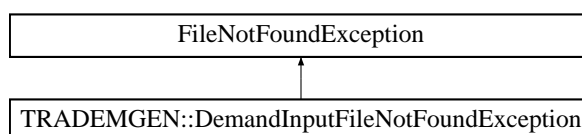
Definition at line 35 of file [FacTRADEMGENServiceContext.cpp](#).

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

- trademgen/factory/[FacTRADEMGENServiceContext.hpp](#)
- trademgen/factory/[FacTRADEMGENServiceContext.cpp](#)

24.27 FileNotFoundException Class Reference

Inheritance diagram for FileNotFoundException:



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

- trademgen/[TRADEMGEN_Exceptions.hpp](#)

24.28 TRADEMGEN::FlagSaver Struct Reference

Public Member Functions

- [FlagSaver](#) (std::ostream &oStream)
- [~FlagSaver](#) ()

24.28.1 Detailed Description

Helper singleton structure to store the current formatting flags of any given output stream. The flags are re-set at the structure destruction.

Definition at line 22 of file [BomDisplay.cpp](#).

24.28.2 Constructor & Destructor Documentation

24.28.2.1 TRADEMGEN::FlagSaver::FlagSaver (std::ostream & oStream) [inline]

Constructor.

Definition at line 25 of file [BomDisplay.cpp](#).

24.28.2.2 TRADEMGEN::FlagSaver::~~FlagSaver () [inline]

Destructor.

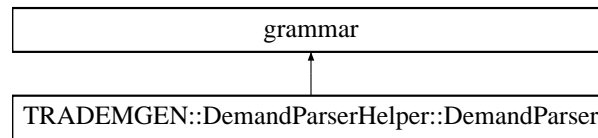
Definition at line 30 of file [BomDisplay.cpp](#).

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

- [trademgen/bom/BomDisplay.cpp](#)

24.29 grammar Class Reference

Inheritance diagram for grammar:



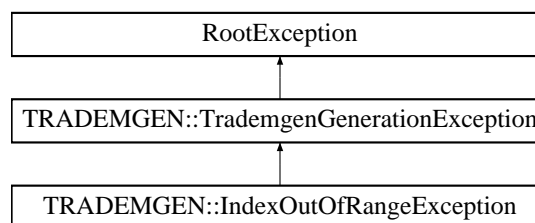
The documentation for this class was generated from the following file:

- [trademgen/command/DemandParserHelper.hpp](#)

24.30 TRADEMGEN::IndexOutOfRangeException Class Reference

```
#include <trademgen/TRADEMGEN_Exceptions.hpp>
```

Inheritance diagram for `TRADEMGEN::IndexOutOfRangeException`:



Public Member Functions

- [IndexOutOfRangeException](#) (const std::string &iWhat)

24.30.1 Detailed Description

Exception when index out of range

Definition at line 43 of file [TRADEMGEN_Exceptions.hpp](#).

24.30.2 Constructor & Destructor Documentation

24.30.2.1 `TRADEMGEN::IndexOutOfRangeException::IndexOutOfRangeException (const std::string &iWhat)` [inline]

Constructor.

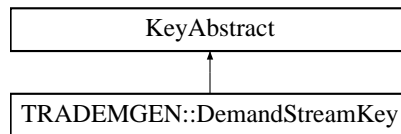
Definition at line 48 of file [TRADEMGEN_Exceptions.hpp](#).

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

- [trademgen/TRADEMGEN_Exceptions.hpp](#)

24.31 KeyAbstract Class Reference

Inheritance diagram for KeyAbstract:



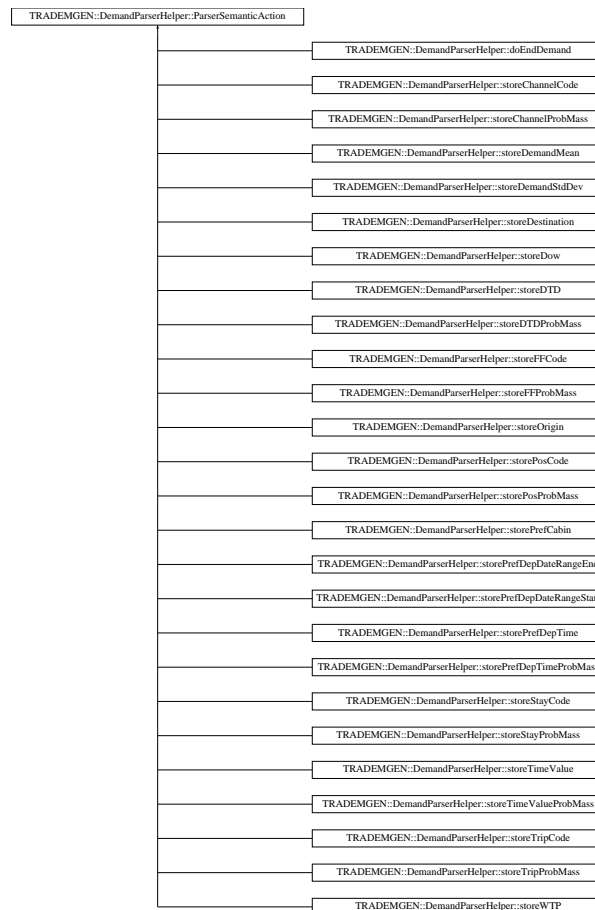
The documentation for this class was generated from the following file:

- trademgen/bom/[DemandStreamKey.hpp](#)

24.32 TRADEMGEN::DemandParserHelper::ParserSemanticAction Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::ParserSemanticAction:



Public Member Functions

- [ParserSemanticAction](#) ([DemandStruct](#) &)

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.32.1 Detailed Description

Generic Semantic Action (Actor / Functor) for the Demand Parser.

Definition at line 30 of file [DemandParserHelper.hpp](#).

24.32.2 Constructor & Destructor Documentation

24.32.2.1 TRADEMGEN::DemandParserHelper::ParserSemanticAction::ParserSemanticAction (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 27 of file [DemandParserHelper.cpp](#).

24.32.3 Member Data Documentation

24.32.3.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMG-EN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMG-EN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMG-EN::DemandParserHelper::doEndDemand::operator\(\)](#).

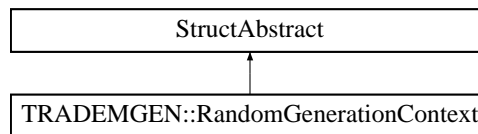
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.33 TRADEMGEN::RandomGenerationContext Struct Reference

```
#include <trademgen/basic/RandomGenerationContext.hpp>
```

Inheritance diagram for TRADEMGEN::RandomGenerationContext:



Public Member Functions

- const stdair::Count_T & [getNumberOfRequestsGeneratedSoFar](#) () const
- const stdair::Probability_T & [getCumulativeProbabilitySoFar](#) () const
- void [setNumberOfRequestsGeneratedSoFar](#) (const stdair::Count_T &iCount)
- void [setCumulativeProbabilitySoFar](#) (const stdair::Probability_T &iProb)
- [RandomGenerationContext](#) ()
- [RandomGenerationContext](#) (const [RandomGenerationContext](#) &)
- [~RandomGenerationContext](#) ()
- void [incrementGeneratedRequestsCounter](#) ()
- void [reset](#) ()
- const std::string [describe](#) () const

24.33.1 Detailed Description

Structure holding the context necessary for demand random generation.

Definition at line 20 of file [RandomGenerationContext.hpp](#).

24.33.2 Constructor & Destructor Documentation

24.33.2.1 TRADEMGEN::RandomGenerationContext::RandomGenerationContext ()

Default constructor.

Definition at line 13 of file [RandomGenerationContext.cpp](#).

24.33.2.2 TRADEMGEN::RandomGenerationContext::RandomGenerationContext (const RandomGenerationContext & iRGC)

Default constructors.

Definition at line 20 of file [RandomGenerationContext.cpp](#).

24.33.2.3 TRADEMGEN::RandomGenerationContext::~~RandomGenerationContext ()

Destructor.

Definition at line 26 of file [RandomGenerationContext.cpp](#).

24.33.3 Member Function Documentation

24.33.3.1 const stdair::Count_T& TRADEMGEN::RandomGenerationContext::getNumberOfRequestsGeneratedSoFar () const [inline]

Get the number of requests generated so far.

Definition at line 26 of file [RandomGenerationContext.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#), [TRADEMGENT::DemandStream::getNumberOfRequestsGeneratedSoFar\(\)](#), and [TRADEMGENT::DemandStream::stillHavingRequestsToBeGenerated\(\)](#).

24.33.3.2 `const stdair::Probability_T& TRADEMGEN::RandomGenerationContext::getCumulativeProbabilitySoFar () const [inline]`

Get the cumulative probability in arrival pattern for last request generated so far (needed for sequential generation).

Definition at line 34 of file [RandomGenerationContext.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#).

24.33.3.3 `void TRADEMGENT::RandomGenerationContext::setNumberOfRequestsGeneratedSoFar (const stdair::Count_T & iCount) [inline]`

Set the number of requests generated so far.

Definition at line 43 of file [RandomGenerationContext.hpp](#).

Referenced by [TRADEMGENT::DemandStream::setNumberOfRequestsGeneratedSoFar\(\)](#).

24.33.3.4 `void TRADEMGENT::RandomGenerationContext::setCumulativeProbabilitySoFar (const stdair::Probability_T & iProb) [inline]`

Set the cumulative probability in arrival pattern for last request generated so far (needed for sequential generation).

Definition at line 51 of file [RandomGenerationContext.hpp](#).

Referenced by [TRADEMGENT::DemandStream::generateTimeOfRequestStatisticsOrder\(\)](#).

24.33.3.5 `void TRADEMGENT::RandomGenerationContext::incrementGeneratedRequestsCounter ()`

Increment counter of requests generated so far.

Definition at line 38 of file [RandomGenerationContext.cpp](#).

Referenced by [TRADEMGENT::DemandStream::incrementGeneratedRequestsCounter\(\)](#).

24.33.3.6 `void TRADEMGENT::RandomGenerationContext::reset ()`

Reset the counters.

Definition at line 43 of file [RandomGenerationContext.cpp](#).

Referenced by [TRADEMGENT::DemandStream::reset\(\)](#).

24.33.3.7 `const std::string TRADEMGENT::RandomGenerationContext::describe () const`

Give a description of the structure (for display purposes).

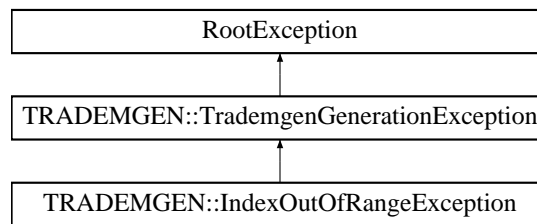
Definition at line 30 of file [RandomGenerationContext.cpp](#).

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

- [trademgen/basic/RandomGenerationContext.hpp](#)
- [trademgen/basic/RandomGenerationContext.cpp](#)

24.34 RootException Class Reference

Inheritance diagram for RootException:

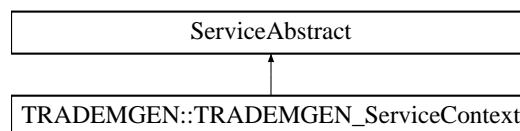


The documentation for this class was generated from the following file:

- trademgen/[TRADEMGEN_Exceptions.hpp](#)

24.35 ServiceAbstract Class Reference

Inheritance diagram for ServiceAbstract:



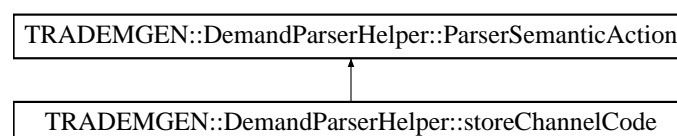
The documentation for this class was generated from the following file:

- trademgen/service/[TRADEMGEN_ServiceContext.hpp](#)

24.36 TRADEMGEN::DemandParserHelper::storeChannelCode Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeChannelCode:



Public Member Functions

- [storeChannelCode](#) ([DemandStruct](#) &)
- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.36.1 Detailed Description

Store the channel type code.

Definition at line 118 of file [DemandParserHelper.hpp](#).

24.36.2 Constructor & Destructor Documentation

24.36.2.1 TRADEMGEN::DemandParserHelper::storeChannelCode::storeChannelCode (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 174 of file [DemandParserHelper.cpp](#).

24.36.3 Member Function Documentation

24.36.3.1 void TRADEMGEN::DemandParserHelper::storeChannelCode::operator() (iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 179 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_itChannelCode](#).

24.36.4 Member Data Documentation

24.36.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

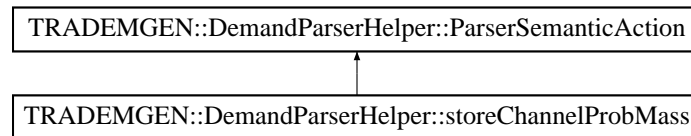
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.37 TRADEMGEN::DemandParserHelper::storeChannelProbMass Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGENT::DemandParserHelper::storeChannelProbMass:



Public Member Functions

- [storeChannelProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.37.1 Detailed Description

Store the channel type probability mass.

Definition at line 126 of file [DemandParserHelper.hpp](#).

24.37.2 Constructor & Destructor Documentation

24.37.2.1 TRADEMGEN::DemandParserHelper::storeChannelProbMass::storeChannelProbMass ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 186 of file [DemandParserHelper.cpp](#).

24.37.3 Member Function Documentation

24.37.3.1 void TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 191 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandStruct::_channelProbDist](#), [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_itChannelCode](#).

24.37.4 Member Data Documentation

24.37.4.1 [DemandStruct](#)& [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#) [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannel-](#)

Code::operator(), operator(), TRADEMGEN::DemandParserHelper::storeTripCode::operator(), TRADEMGEN::DemandParserHelper::storeTripProbMass::operator(), TRADEMGEN::DemandParserHelper::storeStayCode::operator(), TRADEMGEN::DemandParserHelper::storeStayProbMass::operator(), TRADEMGEN::DemandParserHelper::storeFFCode::operator(), TRADEMGEN::DemandParserHelper::storeFFProbMass::operator(), TRADEMGEN::DemandParserHelper::storePrefDepTime::operator(), TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator(), TRADEMGEN::DemandParserHelper::storeWTP::operator(), TRADEMGEN::DemandParserHelper::storeTimeValue::operator(), TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator(), TRADEMGEN::DemandParserHelper::storeDTD::operator(), TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator(), and TRADEMGEN::DemandParserHelper::doEndDemand::operator().

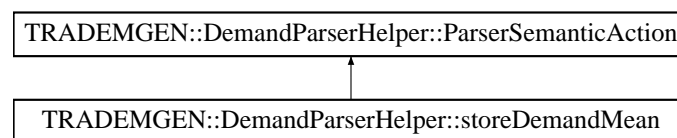
The documentation for this struct was generated from the following files:

- trademgen/command/[DemandParserHelper.hpp](#)
- trademgen/command/[DemandParserHelper.cpp](#)

24.38 TRADEMGEN::DemandParserHelper::storeDemandMean Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDemandMean:



Public Member Functions

- [storeDemandMean](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.38.1 Detailed Description

Store the demand mean value.

Definition at line 86 of file [DemandParserHelper.hpp](#).

24.38.2 Constructor & Destructor Documentation

24.38.2.1 TRADEMGEN::DemandParserHelper::storeDemandMean::storeDemandMean ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 118 of file [DemandParserHelper.cpp](#).

24.38.3 Member Function Documentation

24.38.3.1 void TRADEMGEN::DemandParserHelper::storeDemandMean::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 123 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_demandMean](#).

24.38.4 Member Data Documentation

24.38.4.1 DemandStruct& TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

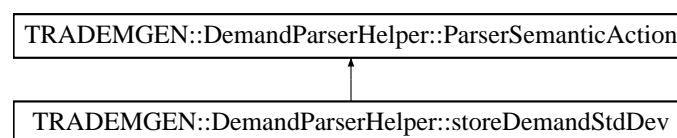
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.39 TRADEMGENT::DemandParserHelper::storeDemandStdDev Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGENT::DemandParserHelper::storeDemandStdDev:



Public Member Functions

- [storeDemandStdDev](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & `_demand`

24.39.1 Detailed Description

Store the demand standard deviation value.

Definition at line 94 of file [DemandParserHelper.hpp](#).

24.39.2 Constructor & Destructor Documentation

24.39.2.1 TRADEMGEN::DemandParserHelper::storeDemandStdDev::storeDemandStdDev (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 129 of file [DemandParserHelper.cpp](#).

24.39.3 Member Function Documentation

24.39.3.1 void TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator() (double iReal) const

Actor Function (functor).

Definition at line 134 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_demandStdDev](#).

24.39.4 Member Data Documentation

24.39.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

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

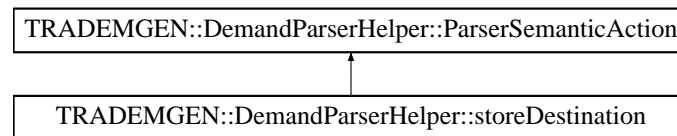
- [trademgen/command/DemandParserHelper.hpp](#)

- [trademgen/command/DemandParserHelper.cpp](#)

24.40 TRADEMGEN::DemandParserHelper::storeDestination Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDestination:



Public Member Functions

- [storeDestination](#) ([DemandStruct](#) &)
- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.40.1 Detailed Description

Store the destination.

Definition at line 70 of file [DemandParserHelper.hpp](#).

24.40.2 Constructor & Destructor Documentation

24.40.2.1 TRADEMGEN::DemandParserHelper::storeDestination::storeDestination ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 93 of file [DemandParserHelper.cpp](#).

24.40.3 Member Function Documentation

24.40.3.1 void TRADEMGEN::DemandParserHelper::storeDestination::operator() ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Actor Function (functor).

Definition at line 98 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEN::DemandStruct::_destination](#).

24.40.4 Member Data Documentation

24.40.4.1 [DemandStruct](#)& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

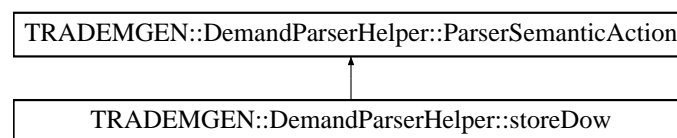
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.41 TRADEMGEN::DemandParserHelper::storeDow Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDow:



Public Member Functions

- [storeDow](#) ([DemandStruct](#) &)
- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.41.1 Detailed Description

Store the DOW (day of the Week).

Definition at line 54 of file [DemandParserHelper.hpp](#).

24.41.2 Constructor & Destructor Documentation

24.41.2.1 TRADEMGEN::DemandParserHelper::storeDow::storeDow (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 71 of file [DemandParserHelper.cpp](#).

24.41.3 Member Function Documentation

24.41.3.1 void TRADEMGEN::DemandParserHelper::storeDow::operator() (iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 76 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_dow](#).

24.41.4 Member Data Documentation

24.41.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand
[inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

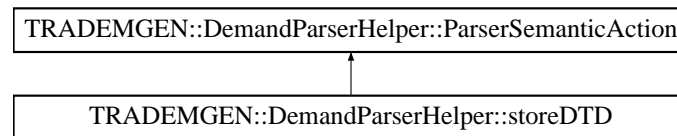
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.42 TRADEMGEN::DemandParserHelper::storeDTD Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGENT::DemandParserHelper::storeDTD:



Public Member Functions

- [storeDTD](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (unsigned int *integer*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.42.1 Detailed Description

Store the parameters for the arrival pattern (as expressed in DTD) continuous probability distribution.

Definition at line 225 of file [DemandParserHelper.hpp](#).

24.42.2 Constructor & Destructor Documentation

24.42.2.1 TRADEMGEN::DemandParserHelper::storeDTD::storeDTD ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 383 of file [DemandParserHelper.cpp](#).

24.42.3 Member Function Documentation

24.42.3.1 void TRADEMGEN::DemandParserHelper::storeDTD::operator() (unsigned int *integer*) const

Actor Function (functor).

Definition at line 388 of file [DemandParserHelper.cpp](#).

References [TRADEMG...::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMG...::DemandStruct::_itDTD](#).

24.42.4 Member Data Documentation

24.42.4.1 [DemandStruct](#)& [TRADEMG...::DemandParserHelper::ParserSemanticAction::_demand](#) [*inherited*]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMG...::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMG...::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMG...::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMG...::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMG...::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMG...::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMG...::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMG...::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMG...::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMG...::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMG...::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMG...::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMG...](#)

[::DemandParserHelper::storeTripCode::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#)(), [operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#)(), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#)).

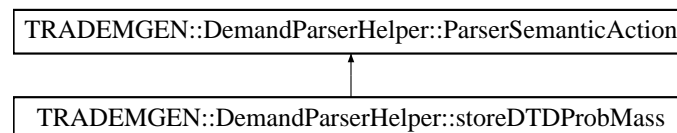
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.43 TRADEMGEN::DemandParserHelper::storeDTDProbMass Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeDTDProbMass:



Public Member Functions

- [storeDTDProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.43.1 Detailed Description

Store the parameters for the arrival pattern (as expressed in DTD) continuous probability distribution.

Definition at line 234 of file [DemandParserHelper.hpp](#).

24.43.2 Constructor & Destructor Documentation

24.43.2.1 TRADEMGEN::DemandParserHelper::storeDTDProbMass::storeDTDProbMass ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 395 of file [DemandParserHelper.cpp](#).

24.43.3 Member Function Documentation

24.43.3.1 void TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 400 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGENT::DemandStruct::_dtdProbDist](#), and [TRADEMGENT::DemandStruct::_itDTD](#).

24.43.4 Member Data Documentation

24.43.4.1 DemandStruct& TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

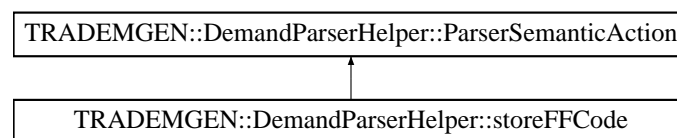
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.44 TRADEMGENT::DemandParserHelper::storeFFCode Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGENT::DemandParserHelper::storeFFCode:



Public Member Functions

- [storeFFCode](#) ([DemandStruct](#) &)
- [void operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.44.1 Detailed Description

Store the frequent flyer code.

Definition at line 166 of file [DemandParserHelper.hpp](#).

24.44.2 Constructor & Destructor Documentation

24.44.2.1 TRADEMGEN::DemandParserHelper::storeFFCode::storeFFCode (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 278 of file [DemandParserHelper.cpp](#).

24.44.3 Member Function Documentation

24.44.3.1 void TRADEMGEN::DemandParserHelper::storeFFCode::operator() (iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 283 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_itFFCode](#).

24.44.4 Member Data Documentation

24.44.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand
[inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)\(\)](#), [operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)\(\)](#).

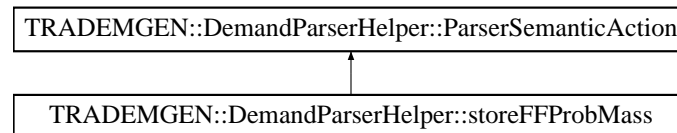
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.45 TRADEMGEN::DemandParserHelper::storeFFProbMass Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeFFProbMass:



Public Member Functions

- [storeFFProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.45.1 Detailed Description

Store the frequent flyer probability mass.

Definition at line 174 of file [DemandParserHelper.hpp](#).

24.45.2 Constructor & Destructor Documentation

24.45.2.1 TRADEMGEN::DemandParserHelper::storeFFProbMass::storeFFProbMass ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 289 of file [DemandParserHelper.cpp](#).

24.45.3 Member Function Documentation

24.45.3.1 void TRADEMGEN::DemandParserHelper::storeFFProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 294 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_ffProbDist](#), and [TRADEMGEN::DemandStruct::_itFFCode](#).

24.45.4 Member Data Documentation

24.45.4.1 [DemandStruct](#)& [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#) [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

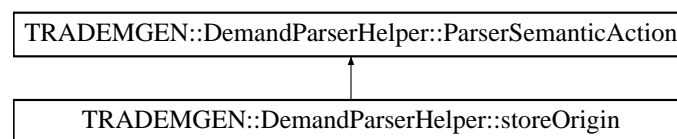
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.46 TRADEMGENT::DemandParserHelper::storeOrigin Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGENT::DemandParserHelper::storeOrigin:



Public Member Functions

- [storeOrigin](#) ([DemandStruct](#) &)
- [void operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.46.1 Detailed Description

Store the origin.

Definition at line 62 of file [DemandParserHelper.hpp](#).

24.46.2 Constructor & Destructor Documentation

24.46.2.1 TRADEMGENT::DemandParserHelper::storeOrigin::storeOrigin ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 82 of file [DemandParserHelper.cpp](#).

24.46.3 Member Function Documentation

24.46.3.1 void TRADEMGEN::DemandParserHelper::storeOrigin::operator()(iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 87 of file [DemandParserHelper.cpp](#).References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_origin](#).

24.46.4 Member Data Documentation

24.46.4.1 DemandStruct& TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand
[inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

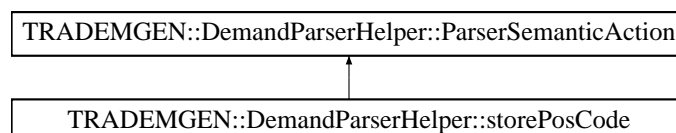
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.47 TRADEMGENT::DemandParserHelper::storePosCode Struct Reference

#include <[trademgen/command/DemandParserHelper.hpp](#)>

Inheritance diagram for TRADEMGENT::DemandParserHelper::storePosCode:



Public Member Functions

- [storePosCode](#) ([DemandStruct](#) &)
- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & `_demand`

24.47.1 Detailed Description

Store the pos type code.

Definition at line 102 of file [DemandParserHelper.hpp](#).

24.47.2 Constructor & Destructor Documentation

24.47.2.1 TRADEMGEN::DemandParserHelper::storePosCode::storePosCode (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 140 of file [DemandParserHelper.cpp](#).

24.47.3 Member Function Documentation

24.47.3.1 void TRADEMGEN::DemandParserHelper::storePosCode::operator() (iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 145 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_itPosCode](#).

24.47.4 Member Data Documentation

24.47.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand
[inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

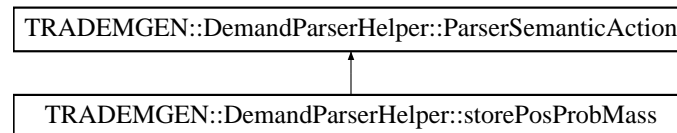
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.48 TRADEMGEN::DemandParserHelper::storePosProbMass Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storePosProbMass:



Public Member Functions

- [storePosProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.48.1 Detailed Description

Store the pos type probability mass.

Definition at line 110 of file [DemandParserHelper.hpp](#).

24.48.2 Constructor & Destructor Documentation

24.48.2.1 TRADEMGEN::DemandParserHelper::storePosProbMass::storePosProbMass ([DemandStruct](#) & [ioDemand](#))

Actor Constructor.

Definition at line 152 of file [DemandParserHelper.cpp](#).

24.48.3 Member Function Documentation

24.48.3.1 void TRADEMGEN::DemandParserHelper::storePosProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 157 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_itPosCode](#), and [TRADEMGEN::DemandStruct::_posProbDist](#).

24.48.4 Member Data Documentation

24.48.4.1 [DemandStruct](#)& [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#) [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

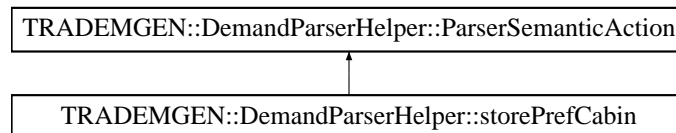
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.49 TRADEMGEN::DemandParserHelper::storePrefCabin Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storePrefCabin:



Public Member Functions

- [storePrefCabin](#) ([DemandStruct](#) &)
- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.49.1 Detailed Description

Store the preferred cabin.

Definition at line 78 of file [DemandParserHelper.hpp](#).

24.49.2 Constructor & Destructor Documentation

24.49.2.1 TRADEMGEN::DemandParserHelper::storePrefCabin::storePrefCabin ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 105 of file [DemandParserHelper.cpp](#).

24.49.3 Member Function Documentation

24.49.3.1 void TRADEMGEN::DemandParserHelper::storePrefCabin::operator() (iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 110 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_prefCabin](#).

24.49.4 Member Data Documentation

24.49.4.1 DemandStruct& TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand
[inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

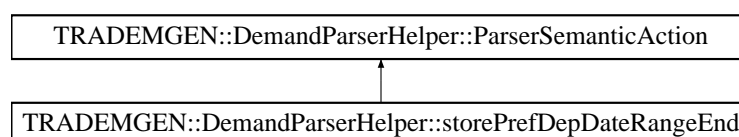
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.50 TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd:



Public Member Functions

- [storePrefDepDateRangeEnd](#) ([DemandStruct](#) &)
- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.50.1 Detailed Description

Store the end of the date range.

Definition at line 46 of file [DemandParserHelper.hpp](#).

24.50.2 Constructor & Destructor Documentation

24.50.2.1 TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::storePrefDepDateRangeEnd ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 48 of file [DemandParserHelper.cpp](#).

24.50.3 Member Function Documentation

24.50.3.1 void TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator() ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Actor Function (functor).

Definition at line 53 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandStruct::_dateRange](#), [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGENT::DemandStruct::_itSeconds](#), [TRADEMGENT::DemandStruct::_prefDepDateEnd](#), [TRADEMGENT::DemandStruct::_prefDepDateStart](#), and [TRADEMGENT::DemandStruct::getDate\(\)](#).

24.50.4 Member Data Documentation

24.50.4.1 [DemandStruct](#)& [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#) [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePref](#)

[DepTimeProbMass::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#)(), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#)(), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#)).

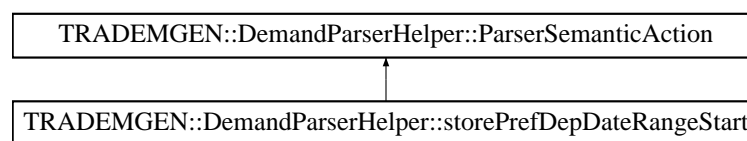
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.51 TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart:



Public Member Functions

- [storePrefDepDateRangeStart](#) ([DemandStruct](#) &)
- [void operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.51.1 Detailed Description

Store the start of the date range.

Definition at line 38 of file [DemandParserHelper.hpp](#).

24.51.2 Constructor & Destructor Documentation

24.51.2.1 TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::storePrefDepDateRangeStart ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 33 of file [DemandParserHelper.cpp](#).

24.51.3 Member Function Documentation

24.51.3.1 void TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator() ([iterator_t](#) *iStr*, [iterator_t](#) *iStrEnd*) const

Actor Function (functor).

Definition at line 38 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_itSeconds](#), [TRADEMGEN::DemandStruct::_prefDepDateStart](#), and [TRADEMGEN::DemandStruct::getDate\(\)](#).

24.51.4 Member Data Documentation

24.51.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#).

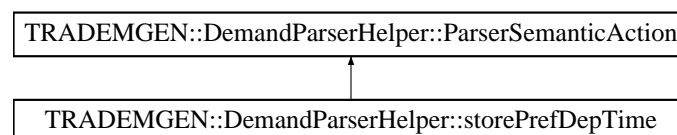
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.52 TRADEMGEN::DemandParserHelper::storePrefDepTime Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storePrefDepTime:



Public Member Functions

- [storePrefDepTime](#) ([DemandStruct](#) &)
- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.52.1 Detailed Description

Store the parameters for the preferred departure time continuous probability distribution.

Definition at line 183 of file [DemandParserHelper.hpp](#).

24.52.2 Constructor & Destructor Documentation

24.52.2.1 TRADEMGEN::DemandParserHelper::storePrefDepTime::storePrefDepTime (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 312 of file [DemandParserHelper.cpp](#).

24.52.3 Member Function Documentation

24.52.3.1 void TRADEMGEN::DemandParserHelper::storePrefDepTime::operator() (iterator_t iStr, iterator_t iStrEnd) const

Actor Function (functor).

Definition at line 317 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGENT::DemandStruct::_itMinutes](#), [TRADEMGENT::DemandStruct::_itPrefDepTime](#), [TRADEMGENT::DemandStruct::_itSeconds](#), and [TRADEMGENT::DemandStruct::getTime\(\)](#).

24.52.4 Member Data Documentation

24.52.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

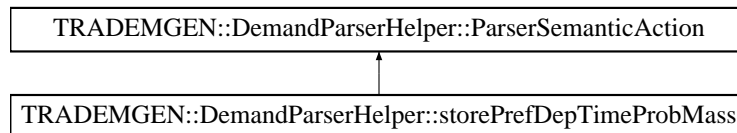
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.53 TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass:



Public Member Functions

- [storePrefDepTimeProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.53.1 Detailed Description

Store the parameters for the preferred departure time continuous probability distribution.

Definition at line 192 of file [DemandParserHelper.hpp](#).

24.53.2 Constructor & Destructor Documentation

24.53.2.1 TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::storePrefDepTimeProbMass ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 332 of file [DemandParserHelper.cpp](#).

24.53.3 Member Function Documentation

24.53.3.1 void TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 337 of file [DemandParserHelper.cpp](#).

References [TRADEMG...DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMG...DemandStruct::_itPrefDepTime](#), and [TRADEMG...DemandStruct::_prefDepTimeProbDist](#).

24.53.4 Member Data Documentation

24.53.4.1 [DemandStruct](#)& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

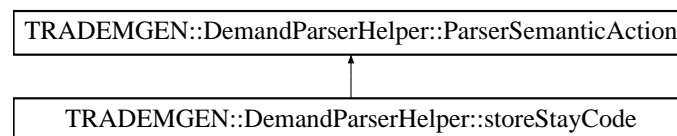
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.54 TRADEMGENT::DemandParserHelper::storeStayCode Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGENT::DemandParserHelper::storeStayCode:



Public Member Functions

- [storeStayCode](#) ([DemandStruct](#) &)
- [void operator\(\)](#) (unsigned int iInteger) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.54.1 Detailed Description

Store the stay type code.

Definition at line 150 of file [DemandParserHelper.hpp](#).

24.54.2 Constructor & Destructor Documentation

24.54.2.1 TRADEMGENT::DemandParserHelper::storeStayCode::storeStayCode ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 244 of file [DemandParserHelper.cpp](#).

24.54.3 Member Function Documentation

24.54.3.1 void TRADEMGEN::DemandParserHelper::storeStayCode::operator() (unsigned int *integer*) const

Actor Function (functor).

Definition at line 249 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_itStayDuration](#).

24.54.4 Member Data Documentation

24.54.4.1 DemandStruct& TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand
[inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDT-D::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

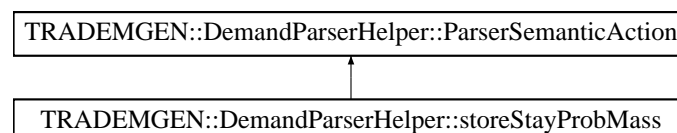
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.55 TRADEMGENT::DemandParserHelper::storeStayProbMass Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGENT::DemandParserHelper::storeStayProbMass:



Public Member Functions

- [storeStayProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & `_demand`

24.55.1 Detailed Description

Store the stay type probability mass.

Definition at line 158 of file [DemandParserHelper.hpp](#).

24.55.2 Constructor & Destructor Documentation

24.55.2.1 TRADEMGEN::DemandParserHelper::storeStayProbMass::storeStayProbMass (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 256 of file [DemandParserHelper.cpp](#).

24.55.3 Member Function Documentation

24.55.3.1 void TRADEMGEN::DemandParserHelper::storeStayProbMass::operator() (double iReal) const

Actor Function (functor).

Definition at line 261 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGENT::DemandStruct::_itStayDuration](#), and [TRADEMGENT::DemandStruct::_stayProbDist](#).

24.55.4 Member Data Documentation

24.55.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)\(\)](#), [operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)\(\)](#).

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

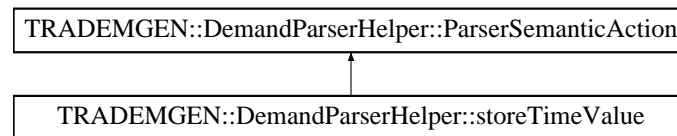
- [trademgen/command/DemandParserHelper.hpp](#)

- [trademgen/command/DemandParserHelper.cpp](#)

24.56 TRADEMGEN::DemandParserHelper::storeTimeValue Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeTimeValue:



Public Member Functions

- [storeTimeValue](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.56.1 Detailed Description

Store the time value.

Definition at line 208 of file [DemandParserHelper.hpp](#).

24.56.2 Constructor & Destructor Documentation

24.56.2.1 TRADEMGEN::DemandParserHelper::storeTimeValue::storeTimeValue ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 359 of file [DemandParserHelper.cpp](#).

24.56.3 Member Function Documentation

24.56.3.1 void TRADEMGEN::DemandParserHelper::storeTimeValue::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 364 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGEN::DemandStruct::_itTimeValue](#).

24.56.4 Member Data Documentation

24.56.4.1 [DemandStruct](#)& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

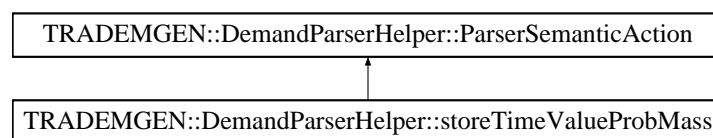
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.57 TRADEMGEN::DemandParserHelper::storeTimeValueProbMass Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeTimeValueProbMass:



Public Member Functions

- [storeTimeValueProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.57.1 Detailed Description

Store the time value probability mass.

Definition at line 216 of file [DemandParserHelper.hpp](#).

24.57.2 Constructor & Destructor Documentation

24.57.2.1 TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::storeTimeValueProbMass (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 370 of file [DemandParserHelper.cpp](#).

24.57.3 Member Function Documentation

24.57.3.1 void TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator() (double iReal) const

Actor Function (functor).

Definition at line 375 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGENT::DemandStruct::_itTimeValue](#), and [TRADEMGENT::DemandStruct::_timeValueProbDist](#).

24.57.4 Member Data Documentation

24.57.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

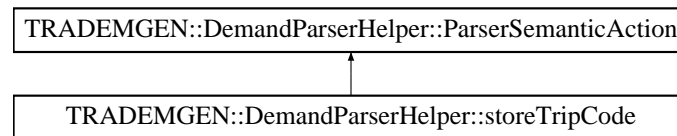
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.58 TRADEMGEN::DemandParserHelper::storeTripCode Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeTripCode:



Public Member Functions

- [storeTripCode](#) ([DemandStruct](#) &)
- void [operator\(\)](#) ([iterator_t](#) iStr, [iterator_t](#) iStrEnd) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.58.1 Detailed Description

Store the trip type code.

Definition at line 134 of file [DemandParserHelper.hpp](#).

24.58.2 Constructor & Destructor Documentation

24.58.2.1 TRADEMGEN::DemandParserHelper::storeTripCode::storeTripCode ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 209 of file [DemandParserHelper.cpp](#).

24.58.3 Member Function Documentation

24.58.3.1 void TRADEMGEN::DemandParserHelper::storeTripCode::operator() ([iterator_t](#) *iStr*, [iterator_t](#) *iStrEnd*) const

Actor Function (functor).

Definition at line 214 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_itTripCode](#).

24.58.4 Member Data Documentation

24.58.4.1 [DemandStruct](#)& [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#) [*inherited*]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [opera-](#)

tor()), TRADEMGEN::DemandParserHelper::storeTripProbMass::operator()), TRADEMGEN::DemandParserHelper::storeStayCode::operator()), TRADEMGEN::DemandParserHelper::storeStayProbMass::operator()), TRADEMGEN::DemandParserHelper::storeFFCode::operator()), TRADEMGEN::DemandParserHelper::storeFFProbMass::operator()), TRADEMGEN::DemandParserHelper::storePrefDepTime::operator()), TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator()), TRADEMGEN::DemandParserHelper::storeWTP::operator()), TRADEMGEN::DemandParserHelper::storeTimeValue::operator()), TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator()), TRADEMGEN::DemandParserHelper::storeDTD::operator()), TRADEMGEN::DemandParserHelper::storeDTPProbMass::operator()), and TRADEMGEN::DemandParserHelper::doEndDemand::operator()).

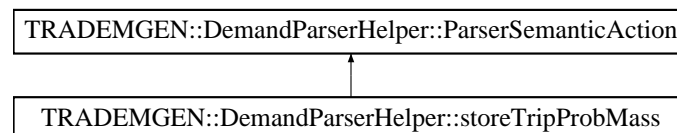
The documentation for this struct was generated from the following files:

- trademgen/command/DemandParserHelper.hpp
- trademgen/command/DemandParserHelper.cpp

24.59 TRADEMGEN::DemandParserHelper::storeTripProbMass Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeTripProbMass:



Public Member Functions

- [storeTripProbMass](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double *iReal*) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.59.1 Detailed Description

Store the trip type probability mass.

Definition at line 142 of file [DemandParserHelper.hpp](#).

24.59.2 Constructor & Destructor Documentation

24.59.2.1 TRADEMGEN::DemandParserHelper::storeTripProbMass::storeTripProbMass ([DemandStruct](#) & *ioDemand*)

Actor Constructor.

Definition at line 221 of file [DemandParserHelper.cpp](#).

24.59.3 Member Function Documentation

24.59.3.1 void TRADEMGEN::DemandParserHelper::storeTripProbMass::operator() (double *iReal*) const

Actor Function (functor).

Definition at line 226 of file [DemandParserHelper.cpp](#).

References [TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand](#), [TRADEMGEN::DemandStruct::_itTripCode](#), and [TRADEMGEN::DemandStruct::_tripProbDist](#).

24.59.4 Member Data Documentation

24.59.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

Referenced by [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTripCode::operator\(\)](#), [operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeWTP::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDT-D::operator\(\)](#), [TRADEMGEN::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGEN::DemandParserHelper::doEndDemand::operator\(\)](#).

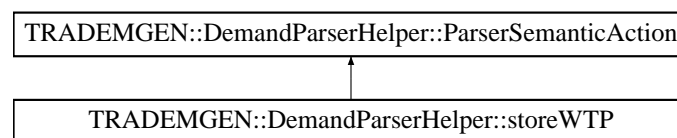
The documentation for this struct was generated from the following files:

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.60 TRADEMGEN::DemandParserHelper::storeWTP Struct Reference

```
#include <trademgen/command/DemandParserHelper.hpp>
```

Inheritance diagram for TRADEMGEN::DemandParserHelper::storeWTP:



Public Member Functions

- [storeWTP](#) ([DemandStruct](#) &)
- void [operator\(\)](#) (double iReal) const

Public Attributes

- [DemandStruct](#) & [_demand](#)

24.60.1 Detailed Description

Store the parameters for the min Willingness-To-Pay (WTP).

Definition at line 200 of file [DemandParserHelper.hpp](#).

24.60.2 Constructor & Destructor Documentation

24.60.2.1 TRADEMGEN::DemandParserHelper::storeWTP::storeWTP (DemandStruct & ioDemand)

Actor Constructor.

Definition at line 348 of file [DemandParserHelper.cpp](#).

24.60.3 Member Function Documentation

24.60.3.1 void TRADEMGEN::DemandParserHelper::storeWTP::operator() (double iReal) const

Actor Function (functor).

Definition at line 353 of file [DemandParserHelper.cpp](#).

References [TRADEMGENT::DemandParserHelper::ParserSemanticAction::_demand](#), and [TRADEMGENT::DemandStruct::_minWTP](#).

24.60.4 Member Data Documentation

24.60.4.1 DemandStruct& TRADEMGEN::DemandParserHelper::ParserSemanticAction::_demand [inherited]

Actor Context.

Definition at line 34 of file [DemandParserHelper.hpp](#).

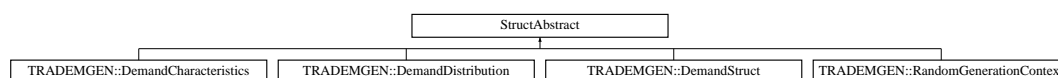
Referenced by [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDow::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeOrigin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDestination::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefCabin::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandMean::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDemandStdDev::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePosProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeChannelProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTripProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeStayProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFCode::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeFFProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTime::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storePrefDepTimeProbMass::operator\(\)](#), [operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValue::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeTimeValueProbMass::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTD::operator\(\)](#), [TRADEMGENT::DemandParserHelper::storeDTDProbMass::operator\(\)](#), and [TRADEMGENT::DemandParserHelper::doEndDemand::operator\(\)](#).

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

- [trademgen/command/DemandParserHelper.hpp](#)
- [trademgen/command/DemandParserHelper.cpp](#)

24.61 StructAbstract Class Reference

Inheritance diagram for StructAbstract:

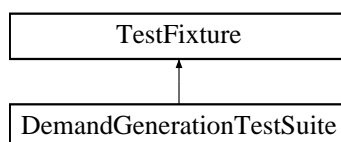


The documentation for this class was generated from the following file:

- trademgen/basic/[RandomGenerationContext.hpp](#)

24.62 TestFixture Class Reference

Inheritance diagram for TestFixture:



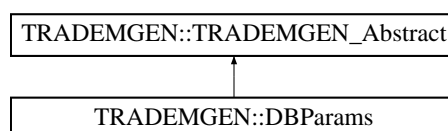
The documentation for this class was generated from the following file:

- test/trademgen/[DemandGenerationTestSuite.hpp](#)

24.63 TRADEMGEN::TRADEMGEM_Abstract Struct Reference

```
#include <trademgen/TRADEMGEM_Abstract.hpp>
```

Inheritance diagram for TRADEMGEN::TRADEMGEM_Abstract:



Public Member Functions

- virtual void [toStream](#) (std::ostream &ioOut) const =0
- virtual void [fromStream](#) (std::istream &ioIn)=0
- virtual std::string [toString](#) () const =0

Protected Member Functions

- [TRADEMGEM_Abstract](#) ()
- [TRADEMGEM_Abstract](#) (const [TRADEMGEM_Abstract](#) &)
- virtual [~TRADEMGEM_Abstract](#) ()

24.63.1 Detailed Description

Base class for the [TRADEMGEN](#) interface structures.

Definition at line 16 of file [TRADEMGEN_Abstract.hpp](#).

24.63.2 Constructor & Destructor Documentation

24.63.2.1 TRADEMGEN::TRADEMGEN_Abstract::TRADEMGEN_Abstract () [inline, protected]

Protected Default Constructor to ensure this class is abstract.

Definition at line 33 of file [TRADEMGEN_Abstract.hpp](#).

24.63.2.2 TRADEMGEN::TRADEMGEN_Abstract::TRADEMGEN_Abstract (const TRADEMGEN_Abstract &) [inline, protected]

Definition at line 34 of file [TRADEMGEN_Abstract.hpp](#).

24.63.2.3 virtual TRADEMGEN::TRADEMGEN_Abstract::~~TRADEMGEN_Abstract () [inline, protected, virtual]

Destructor.

Definition at line 37 of file [TRADEMGEN_Abstract.hpp](#).

24.63.3 Member Function Documentation

24.63.3.1 virtual void TRADEMGEN::TRADEMGEN_Abstract::toStream (std::ostream & ioOut) const [pure virtual]

Dump a structure into an output stream.

Parameters

<i>ostream&</i>	the output stream.
---------------------	--------------------

Implemented in [TRADEMGEN::DBParams](#).

24.63.3.2 virtual void TRADEMGEN::TRADEMGEN_Abstract::fromStream (std::istream & ioIn) [pure virtual]

Read a structure from an input stream.

Parameters

<i>istream&</i>	the input stream.
---------------------	-------------------

Implemented in [TRADEMGEN::DBParams](#).

Referenced by [operator>>\(\)](#).

24.63.3.3 virtual std::string TRADEMGEN::TRADEMGEN_Abstract::toString () const [pure virtual]

Get the serialised version of the structure.

Implemented in [TRADEMGEN::DBParams](#).

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

- trademgen/[TRADEMGEN_Abstract.hpp](#)

24.64 TRADEMGEN::TRADEMGEN_Service Class Reference

class holding the services related to Travel Demand Generation.

```
#include <trademgen/TRADEMGEN_Service.hpp>
```

Public Member Functions

- [TRADEMGEN_Service](#) (const stdair::BasLogParams &, const stdair::BasDBParams &, const stdair::RandomSeed_T &)
Constructor.
- [TRADEMGEN_Service](#) (const stdair::BasLogParams &, const stdair::RandomSeed_T &)
- [TRADEMGEN_Service](#) (stdair::STDAIR_ServicePtr_T, const stdair::RandomSeed_T &)
- void [parseAndLoad](#) (const stdair::Filename_T &iDemandInputFilename)
- [~TRADEMGEN_Service](#) ()
- void [buildSampleBom](#) ()
- stdair::BookingRequestStruct [buildSampleBookingRequest](#) (const bool isForCRS=false)
- void [displayAirlineListFromDB](#) () const
- const stdair::Count_T & [getExpectedTotalNumberOfRequestsToBeGenerated](#) () const
- const stdair::Count_T & [getActualTotalNumberOfRequestsToBeGenerated](#) () const
- const bool [stillHavingRequestsToBeGenerated](#) (const stdair::DemandStreamKeyStr_T &, stdair::Progress-StatusSet &, const stdair::DemandGenerationMethod &) const
- stdair::Count_T [generateFirstRequests](#) (const stdair::DemandGenerationMethod &) const
- stdair::BookingRequestPtr_T [generateNextRequest](#) (const stdair::DemandStreamKeyStr_T &, const stdair::DemandGenerationMethod &) const
- stdair::ProgressStatusSet [popEvent](#) (stdair::EventStruct &) const
- bool [isQueueDone](#) () const
- bool [generateCancellation](#) (const stdair::TravelSolutionStruct &, const stdair::PartySize_T &, const stdair::DateTime_T &, const stdair::Date_T &) const
- void [reset](#) () const
- std::string [csvDisplay](#) () const

24.64.1 Detailed Description

class holding the services related to Travel Demand Generation.

Definition at line 38 of file [TRADEMGEN_Service.hpp](#).

24.64.2 Constructor & Destructor Documentation

24.64.2.1 TRADEMGEN::TRADEMGEN_Service::TRADEMGEN_Service (const stdair::BasLogParams & iLogParams, const stdair::BasDBParams & iDBParams, const stdair::RandomSeed_T & iRandomSeed)

Constructor.

The initTrademgenService() method is called; see the corresponding documentation for more details.

A reference on an output stream is given, so that log outputs can be directed onto that stream.

Moreover, database connection parameters are given, so that a session can be created on the corresponding database.

Parameters

<i>const</i>	stdair::BasLogParams& Parameters for the output log stream.
<i>const</i>	stdair::BasDBParams& Parameters for the database access.
<i>const</i>	stdair::RandomSeed_T& Seed for the random generation.

Definition at line 71 of file [TRADEMGEN_Service.cpp](#).

24.64.2.2 TRADEMGEN::TRADEMGEN_Service::TRADEMGEN_Service (const stdair::BasLogParams & iLogParams, const stdair::RandomSeed_T & iRandomSeed)

Constructor.

The initTrademgenService() method is called; see the corresponding documentation for more details.

A reference on an output stream is given, so that log outputs can be directed onto that stream.

Parameters

<i>const</i>	stdair::BasLogParams& Parameters for the output log stream.
<i>const</i>	stdair::RandomSeed_T& Seed for the random generation.

Definition at line 50 of file [TRADEMGEN_Service.cpp](#).

24.64.2.3 TRADEMGEN::TRADEMGEN_Service::TRADEMGEN_Service (stdair::STDAIR_ServicePtr_T ioSTDAIR_Service_ptr, const stdair::RandomSeed_T & iRandomSeed)

Constructor.

The initTrademgenService() method is called; see the corresponding documentation for more details.

Moreover, as no reference on any output stream is given, neither any database access parameter is given, it is assumed that the StdAir log service has already been initialised with the proper log output stream by some other methods in the calling chain (for instance, when the [TRADEMGEN_Service](#) is itself being initialised by another library service such as DSIM_Service).

Parameters

<i>stdair::STDAIR_ServicePtr_T</i>	Handler on the STDAIR_Service.
<i>const</i>	stdair::RandomSeed_T& Seed for the random generation.

Definition at line 94 of file [TRADEMGEN_Service.cpp](#).

24.64.2.4 TRADEMGEN::TRADEMGEN_Service::~~TRADEMGEN_Service ()

Destructor.

Definition at line 111 of file [TRADEMGEN_Service.cpp](#).

24.64.3 Member Function Documentation

24.64.3.1 void TRADEMGEN::TRADEMGEN_Service::parseAndLoad (const stdair::Filename_T & iDemandInputFilename)

Parse the demand input file.

The CSV file, describing the parameters of the demand to be generated for the simulator, is parsed and instantiated in memory accordingly.

Parameters

<i>const</i>	stdair::Filename_T& Filename of the input demand file.
--------------	--

Definition at line 187 of file [TRADEMGEN_Service.cpp](#).

References [generateDemand\(\)](#).

Referenced by [TRADEMGEN::Trademgener::init\(\)](#), and [main\(\)](#).

24.64.3.2 void TRADEMGEN::TRADEMGEN_Service::buildSampleBom ()

Build a sample BOM tree, made of a single [DemandStream](#) object.

As of now (March 2011), it corresponds to:

- Origin: SIN
- Destination: BKK
- Preferred departure date: 2011-02-14
- Preferred cabin: Y (Economy)
- POS distribution:
 - BKK: 30%
 - SIN: 70%
- Channel distribution:
 - Direct Offline: 10%
 - Direct Online: 30%
 - Indirect Offline: 40%
 - Indirect Online: 20%
- Trip type distribution:
 - Outbound: 60%
 - Inbound: 20%
 - One-way: 20%
- Arrival pattern distribution:
 - 330 DTD: 0%
 - 40 DTD: 20%
 - 20 DTD: 60%
 - 1 DTD: 100%

15:0, 60:1
- Stay duration distribution:
 - 0 day: 10%
 - 1 day: 10%
 - 2 days: 15%
 - 3 days: 15%
 - 4 days: 15%
 - 5 days: 35%
- Frequent flyer distribution:
 - Platinum: 1%
 - Gold: 5%
 - Silver: 15%
 - Member: 30%
 - No card: 49%
- Preferred departure time (cumulative distribution):

- 6am: 0%
- 7am: 10%
- 9am: 30%
- 5pm: 40%
- 7pm: 80%
- 8pm: 95%
- 10pm: 100%
- Value of time distribution:
 - 15 min: 0%
 - 60 min: 100%
- WTP: 200
- Number of requests: Normal (mu = 10.0, std_dev = 1.0)
- Change fee: 20; Non refundable; Saturday night stay

Definition at line 220 of file [TRADEMGEN_Service.cpp](#).

Referenced by [main\(\)](#).

24.64.3.3 `stdair::BookingRequestStruct TRADEMGEN::TRADEMGEN_Service::buildSampleBookingRequest (const bool isForCRS = false)`

Build a sample booking request structure.

As of now (March 2011), the sample booking request is made of the following parameters:

- Return trip (inbound): LHR-SYD (POS: LHR, Channel: DN),
- Departing 10-JUN-2011 around 8:00, staying 7 days
- Requested on 15-MAY-2011 at 10:00
- Economy cabin, 3 persons, FF member
- WTP: 1000.0 EUR
- Dis-utility: 100.0 EUR/hour

As of now (March 2011), the CRS-related booking request is made of the following parameters:

- Return trip (inbound): SIN-BKK (POS: SIN, Channel: IN),
- Departing 30-JAN-2010 around 10:00, staying 7 days
- Requested on 22-JAN-2010 at 10:00
- Economy cabin, 3 persons, FF member
- WTP: 1000.0 EUR
- Dis-utility: 100.0 EUR/hour

See also

`stdair::CmdBomManager` for more details.

Parameters

<i>const</i>	bool isForCRS Whether the sample booking request is for CRS.
--------------	--

Returns

BookingRequestStruct& Sample booking request structure.

Definition at line 279 of file [TRADEMGEN_Service.cpp](#).

24.64.3.4 void TRADEMGEN::TRADEMGEN_Service::displayAirlineListFromDB () const

Display the list of airlines, as held within the sample database.

Definition at line 324 of file [TRADEMGEN_Service.cpp](#).

Referenced by [main\(\)](#), and [TRADEMGEN::Trademgener::trademgen\(\)](#).

24.64.3.5 const stdair::Count_T & TRADEMGEN::TRADEMGEN_Service::getExpectedTotalNumberOfRequestsToBeGenerated () const

Get the expected number of events/requests to be generated for all the demand streams.

The getExpectedTotalNbOfEvents() method is called on the underlying EventQueue object, which keeps track of that number.

Note

That number usually corresponds to an expectation (i.e., the mean value of a random distribution). The actual number will be drawn when calling the [generateFirstRequests\(\)](#) method.

Returns

const stdair::Count_T& Expected number of events to be generated.

Definition at line 385 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#).

24.64.3.6 const stdair::Count_T & TRADEMGEN::TRADEMGEN_Service::getActualTotalNumberOfRequestsToBeGenerated () const

Get the actual number of events/requests to be generated for all the demand streams.

The getActualTotalNbOfEvents() method is called on the underlying EventQueue object, which keeps track of that number.

Note

That number has been drawn when calling the [generateFirstRequests\(\)](#) method.

Returns

const stdair::Count_T& Expected number of events to be generated.

Definition at line 409 of file [TRADEMGEN_Service.cpp](#).

24.64.3.7 const bool TRADEMGEN::TRADEMGEN_Service::stillHavingRequestsToBeGenerated (const stdair::DemandStreamKeyStr_T & iKey, stdair::ProgressStatusSet & ioPSS, const stdair::DemandGenerationMethod & iDemandGenerationMethod) const

Check whether enough requests have already been generated for the demand stream which corresponds to the given key.

Parameters

<i>const</i>	DemandStreamKey & A string identifying uniquely the demand stream (e.g., "SIN-HND 2010--Feb-08 Y").
<i>const</i>	<code>stdair::DemandGenerationMethod</code> & States whether the demand generation must be performed following the method based on statistic orders. The alternative method, while more "intuitive", is also a sequential algorithm.

Returns

`bool` Whether or not there are still events to be generated for that demand stream.

Definition at line 433 of file [TRADEMGEN_Service.cpp](#).

24.64.3.8 `stdair::Count_T TRADEMGEN::TRADEMGEN_Service::generateFirstRequests (const stdair::DemandGenerationMethod & iDemandGenerationMethod) const`

Browse the list of demand streams and generate the first request of each stream.

Parameters

<i>const</i>	<code>stdair::DemandGenerationMethod</code> & States whether the demand generation must be performed following the method based on statistic orders. The alternative method, while more "intuitive", is also a sequential algorithm.
--------------	--

Returns

`stdair::Count_T` The expected total number of events to be generated

Definition at line 460 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#).

24.64.3.9 `stdair::BookingRequestPtr_T TRADEMGEN::TRADEMGEN_Service::generateNextRequest (const stdair::DemandStreamKeyStr_T & iKey, const stdair::DemandGenerationMethod & iDemandGenerationMethod) const`

Generate a request with the demand stream which corresponds to the given key.

Parameters

<i>const</i>	DemandStreamKey & A string identifying uniquely the demand stream (e.g., "SIN-HND 2010--Feb-08 Y").
<i>const</i>	<code>stdair::DemandGenerationMethod</code> & States whether the demand generation must be performed following the method based on statistic orders. The alternative method, while more "intuitive", is also a sequential algorithm.

Returns

`stdair::BookingRequestPtr_T` (Boost) shared pointer on the booking request structure, which has just been created.

Definition at line 489 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#).

24.64.3.10 `stdair::ProgressStatusSet TRADEMGEN::TRADEMGEN_Service::popEvent (stdair::EventStruct & ioEventStruct) const`

Pop the next coming (in time) event, and remove it from the event queue.

- The next coming (in time) event corresponds to the event having the earliest date-time stamp. In other words,

it is the first/front element of the event queue.

- That (first) event/element is then removed from the event queue
- The progress status is updated for the corresponding demand stream.

Returns

stdair::EventStruct A copy of the event structure, which comes first in time from within the event queue.

Definition at line 515 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#).

24.64.3.11 bool TRADEMGEN::TRADEMGEN_Service::isQueueDone () const

States whether the event queue has reached the end.

For now, that method states whether the event queue is empty.

Definition at line 534 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#).

24.64.3.12 bool TRADEMGEN::TRADEMGEN_Service::generateCancellation (const stdair::TravelSolutionStruct & iTravelSolution, const stdair::PartySize_T & iPartySize, const stdair::DateTime_T & iRequestTime, const stdair::Date_T & iDepartureDate) const

Generate the potential cancellation event.

Definition at line 557 of file [TRADEMGEN_Service.cpp](#).

24.64.3.13 void TRADEMGEN::TRADEMGEN_Service::reset () const

Reset the context of the demand streams for another demand generation without having to reparse the demand input file.

Definition at line 584 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#).

24.64.3.14 std::string TRADEMGEN::TRADEMGEN_Service::csvDisplay () const

Recursively display (dump in the returned string) the objects of the BOM tree.

Returns

std::string Output string in which the BOM tree is logged/dumped.

Definition at line 300 of file [TRADEMGEN_Service.cpp](#).

Referenced by [generateDemand\(\)](#).

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

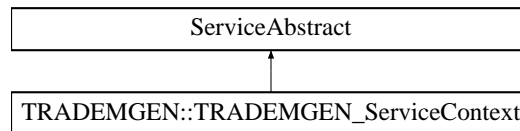
- trademgen/[TRADEMGEN_Service.hpp](#)
- trademgen/service/[TRADEMGEN_Service.cpp](#)

24.65 TRADEMGEN::TRADEMGEN_ServiceContext Class Reference

Class holding the context of the Trademgen services.

```
#include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
```

Inheritance diagram for TRADEMGEN::TRADEMGEN_ServiceContext:



Friends

- class [TRADEMGEN_Service](#)
- class [FacTRADEMGENServiceContext](#)

24.65.1 Detailed Description

Class holding the context of the Trademgen services.

Definition at line 30 of file [TRADEMGEN_ServiceContext.hpp](#).

24.65.2 Friends And Related Function Documentation

24.65.2.1 friend class TRADEMGEN_Service [friend]

The [TRADEMGEN_Service](#) class should be the sole class to get access to ServiceContext content: general users do not want to bother with a context interface.

Definition at line 36 of file [TRADEMGEN_ServiceContext.hpp](#).

24.65.2.2 friend class FacTRADEMGENServiceContext [friend]

Definition at line 37 of file [TRADEMGEN_ServiceContext.hpp](#).

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

- trademgen/service/[TRADEMGEN_ServiceContext.hpp](#)
- trademgen/service/[TRADEMGEN_ServiceContext.cpp](#)

24.66 TRADEMGEN::Trademgener Struct Reference

Public Member Functions

- `std::string trademgen` (const std::string &iQuery)
- `Trademgener` ()
- `Trademgener` (const [Trademgener](#) &iTrademgener)
- `~Trademgener` ()
- `bool init` (const std::string &iLogFilepath, const stdair::RandomSeed_T &iRandomSeed, const stdair::Filename_T &iDemandInputFilename, const std::string &iDBUser, const std::string &iDBPasswd, const std::string &iDBHost, const std::string &iDBPort, const std::string &iDBDBName)

24.66.1 Detailed Description

Definition at line 22 of file [pytrademgen.cpp](#).

24.66.2 Constructor & Destructor Documentation

24.66.2.1 TRADEMGEN::Trademgener::Trademgener () [inline]

Default constructor.

Definition at line 79 of file [pytrademgen.cpp](#).

24.66.2.2 TRADEMGEN::Trademgener::Trademgener (const Trademgener & iTrademgener) [inline]

Default copy constructor.

Definition at line 83 of file [pytrademgen.cpp](#).

24.66.2.3 TRADEMGEN::Trademgener::~~Trademgener () [inline]

Default constructor.

Definition at line 89 of file [pytrademgen.cpp](#).

24.66.3 Member Function Documentation

24.66.3.1 std::string TRADEMGEN::Trademgener::trademgen (const std::string & iQuery) [inline]

Wrapper around the travel demand generation use case.

Definition at line 25 of file [pytrademgen.cpp](#).

References [TRADEMGEN::TRADEMGEN_Service::displayAirlineListFromDB\(\)](#).

Referenced by [BOOST_PYTHON_MODULE\(\)](#).

24.66.3.2 bool TRADEMGEN::Trademgener::init (const std::string & iLogFilepath, const stdair::RandomSeed_T & iRandomSeed, const stdair::Filename_T & iDemandInputFilename, const std::string & iDBUser, const std::string & iDBPasswd, const std::string & iDBHost, const std::string & iDBPort, const std::string & iDBDBName) [inline]

Wrapper around the search use case.

Definition at line 95 of file [pytrademgen.cpp](#).

References [TRADEMGEN::TRADEMGEN_Service::parseAndLoad\(\)](#).

Referenced by [BOOST_PYTHON_MODULE\(\)](#).

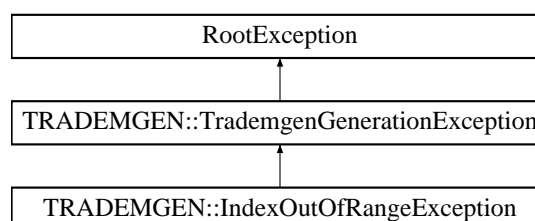
The documentation for this struct was generated from the following file:

- [trademgen/python/pytrademgen.cpp](#)

24.67 TRADEMGEN::TrademgenGenerationException Class Reference

```
#include <trademgen/TRADEMGEN_Exceptions.hpp>
```

Inheritance diagram for TRADEMGEN::TrademgenGenerationException:



Public Member Functions

- [TrademgenGenerationException](#) (const std::string &iWhat)

24.67.1 Detailed Description

Root exception for the TraDemGen component

Definition at line 18 of file [TRADEMGEN_Exceptions.hpp](#).

24.67.2 Constructor & Destructor Documentation

24.67.2.1 TRADEMGEN::TrademgenGenerationException::TrademgenGenerationException (const std::string & iWhat) [inline]

Constructor.

Definition at line 23 of file [TRADEMGEN_Exceptions.hpp](#).

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

- trademgen/[TRADEMGEN_Exceptions.hpp](#)

25 File Documentation

25.1 [doc/local/authors.doc](#) File Reference

25.2 [doc/local/codingrules.doc](#) File Reference

25.3 [doc/local/copyright.doc](#) File Reference

25.4 [doc/local/documentation.doc](#) File Reference

25.5 [doc/local/features.doc](#) File Reference

25.6 [doc/local/help_wanted.doc](#) File Reference

25.7 [doc/local/howto_release.doc](#) File Reference

25.8 [doc/local/index.doc](#) File Reference

25.9 [doc/local/installation.doc](#) File Reference

25.10 [doc/local/linking.doc](#) File Reference

25.11 [doc/local/test.doc](#) File Reference

25.12 [doc/local/users_guide.doc](#) File Reference

25.13 [doc/local/verification.doc](#) File Reference

25.14 [doc/tutorial/tutorial.doc](#) File Reference

25.15 test/trademgen/DemandGenerationTestSuite.cpp File Reference

25.16 DemandGenerationTestSuite.cpp

```

00001
00005 // //////////////////////////////////////
00006 // Import section
00007 // //////////////////////////////////////
00008 // STL
00009 #include <sstream>
00010 #include <fstream>
00011 #include <map>
00012 #include <cmath>
00013 // Boost Unit Test Framework (UTF)
00014 #define BOOST_TEST_DYN_LINK
00015 #define BOOST_TEST_MAIN
00016 #define BOOST_TEST_MODULE DemandGenerationTest
00017 #include <boost/test/unit_test.hpp>
00018 // StdAir
00019 #include <stdair/stdair_basic_types.hpp>
00020 #include <stdair/basic/BasConst_General.hpp>
00021 #include <stdair/basic/BasLogParams.hpp>
00022 #include <stdair/basic/BasDBParams.hpp>
00023 #include <stdair/basic/BasFileMgr.hpp>
00024 #include <stdair/basic/ProgressStatusSet.hpp>
00025 #include <stdair/bom/EventStruct.hpp>
00026 #include <stdair/bom/EventQueue.hpp>
00027 #include <stdair/bom/BookingRequestStruct.hpp>
00028 #include <stdair/service/Logger.hpp>
00029 // TraDemGen
00030 #include <trademgen/TRADEMGEN_Service.hpp>
00031 #include <trademgen/bom/DemandStreamKey.hpp>
00032 #include <trademgen/config/trademgen-paths.hpp>
00033
00034 namespace boost_utf = boost::unit_test;
00035
00036 // (Boost) Unit Test XML Report
00037 std::ofstream utfReportStream ("DemandGenerationTestSuite_utfresults.xml");
00038
00042 struct UnitTestConfig {
00044     UnitTestConfig() {
00045         boost_utf::unit_test_log.set_stream (utfReportStream);
00046         boost_utf::unit_test_log.set_format (boost_utf::XML);
00047         boost_utf::unit_test_log.set_threshold_level (boost_utf::log_test_units);
00048         //boost_utf::unit_test_log.set_threshold_level
00049         (boost_utf::log_successful_tests);
00050     }
00052     ~UnitTestConfig() {
00053     }
00054 };
00055
00056 // Specific type definitions
00057 typedef std::pair<stdair::Count_T, stdair::Count_T> NbOfEventsPair_T;
00058 typedef std::map<const stdair::DemandStreamKeyStr_T,
00059     NbOfEventsPair_T> NbOfEventsByDemandStreamMap_T;
00060
00061
00062 // ////////////////////////////////// Main: Unit Test Suite //////////////////////////////////
00063
00064 // Set the UTF configuration (re-direct the output to a specific file)
00065 BOOST_GLOBAL_FIXTURE (UnitTestConfig);
00066
00067 // Start the test suite
00068 BOOST_AUTO_TEST_SUITE (master_test_suite)
00069
00070
00073 BOOST_AUTO_TEST_CASE (trademgen_simple_simulation_test) {
00074
00075     // Seed for the random generation
00076     const stdair::RandomSeed_T lRandomSeed = stdair::DEFAULT_RANDOM_SEED;
00077
00078     // Input file name
00079     const stdair::Filename_T lInputFilename (STDAIR_SAMPLE_DIR "/demand01.csv");
00080
00081     // Check that the file path given as input corresponds to an actual file
00082     const bool doesExistAndIsReadable =
00083         stdair::BasFileMgr::doesExistAndIsReadable (lInputFilename);
00084     BOOST_CHECK_MESSAGE (doesExistAndIsReadable == true,
00085         "The '" << lInputFilename
00086         << "' input file can not be open and read");
00087
00088     // Output log File
00089     const stdair::Filename_T lLogFilename ("DemandGenerationTestSuite.log");
00090

```



```

00091 // Set the log parameters
00092 std::ofstream logOutputFile;
00093 // open and clean the log outputfile
00094 logOutputFile.open (lLogFilename.c_str());
00095 logOutputFile.clear();
00096
00097 // Initialise the TraDemGen service object
00098 const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
00099 TRADEMGEN::TRADEMGEN_Service trademgenService (lLogParams, lRandomSeed);
00100
00101 // Create the DemandStream objects, and insert them within the BOM tree
00102 BOOST_CHECK_NO_THROW (trademgenService.parseAndLoad (lInputFilename));
00103
00113 NbOfEventsByDemandStreamMap_T lNbOfEventsMap;
00114 lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00115     value_type ("SIN-HND 2010-Feb-08 Y",
00116         NbOfEventsPair_T (1, 10)));
00117 lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00118     value_type ("SIN-HND 2010-Feb-09 Y",
00119         NbOfEventsPair_T (1, 10)));
00120 lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00121     value_type ("SIN-BKK 2010-Feb-08 Y",
00122         NbOfEventsPair_T (1, 10)));
00123 lNbOfEventsMap.insert (NbOfEventsByDemandStreamMap_T::
00124     value_type ("SIN-BKK 2010-Feb-09 Y",
00125         NbOfEventsPair_T (1, 10)));
00126 // Total number of events, for all the demand streams: 3
00127 stdair::Count_T lRefExpectedNbOfEvents (40);
00128
00129 // Retrieve the expected (mean value of the) number of events to be
00130 // generated
00131 const stdair::Count_T& lExpectedNbOfEventsToBeGenerated =
00132     trademgenService.getExpectedTotalNumberOfRequestsToBeGenerated();
00133
00134 BOOST_CHECK_EQUAL (lRefExpectedNbOfEvents,
00135     std::floor (lExpectedNbOfEventsToBeGenerated));
00136
00137 BOOST_CHECK_MESSAGE (lRefExpectedNbOfEvents ==
00138     std::floor (lExpectedNbOfEventsToBeGenerated),
00139     "Expected total number of requests to be generated: "
00140     << lExpectedNbOfEventsToBeGenerated
00141     << " (=> "
00142     << std::floor (lExpectedNbOfEventsToBeGenerated)
00143     << "). Reference value: " << lRefExpectedNbOfEvents);
00144
00145 // Generate the date time of the requests with the statistic order method.
00146 stdair::DemandGenerationMethod lDemandGenerationMethod (
    stdair::DemandGenerationMethod::STA_ORD);
00147
00157 const stdair::Count_T& lActualNbOfEventsToBeGenerated =
00158     trademgenService.generateFirstRequests (lDemandGenerationMethod);
00159
00160 // DEBUG
00161 STDAIR_LOG_DEBUG ("Expected number of events: "
00162     << lExpectedNbOfEventsToBeGenerated << ", actual: "
00163     << lActualNbOfEventsToBeGenerated);
00164
00165 // Total number of events, for all the demand streams: 40
00166 const stdair::Count_T lRefActualNbOfEvents (40);
00167 BOOST_CHECK_EQUAL (lRefActualNbOfEvents, lActualNbOfEventsToBeGenerated);
00168
00169 BOOST_CHECK_MESSAGE (lRefActualNbOfEvents == lActualNbOfEventsToBeGenerated,
00170     "Actual total number of requests to be generated: "
00171     << lExpectedNbOfEventsToBeGenerated
00172     << " (=> "
00173     << std::floor (lExpectedNbOfEventsToBeGenerated)
00174     << "). Reference value: " << lRefActualNbOfEvents);
00175
00177 const bool isQueueDone = trademgenService.isQueueDone();
00178 BOOST_REQUIRE_MESSAGE (isQueueDone == false,
00179     "The event queue should not be empty. You may check "
00180     << "the input file: '" << lInputFilename << "'");
00181
00189 stdair::Count_T idx = 1;
00190 while (trademgenService.isQueueDone() == false) {
00191
00192     // Get the next event from the event queue
00193     stdair::EventStruct lEventStruct;
00194     stdair::ProgressStatusSet lPPS = trademgenService.popEvent (lEventStruct);
00195
00196     // DEBUG
00197     STDAIR_LOG_DEBUG ("Popped event: '" << lEventStruct.describe() << "'");
00198
00199     // Extract the corresponding demand/booking request
00200     const stdair::BookingRequestStruct& lPoppedRequest =
00201         lEventStruct.getBookingRequest();
00202

```

```

00203 // DEBUG
00204 STDAIR_LOG_DEBUG ("Popped booking request: '"
00205                  << lPoppedRequest.describe() << "'.");
00206
00207 // Retrieve the corresponding demand stream
00208 const stdair::DemandGeneratorKey_T& lDemandStreamKey =
00209     lPoppedRequest.getDemandGeneratorKey();
00210
00211 // Check that the number of booking requests to be generated are correct
00212 const NbOfEventsByDemandStreamMap_T::iterator itNbOfEventsMap =
00213     lNbOfEventsMap.find (lDemandStreamKey);
00214 BOOST_REQUIRE_MESSAGE (itNbOfEventsMap != lNbOfEventsMap.end(),
00215     "The demand stream key '" << lDemandStreamKey
00216     << "' is not expected in that test");
00217
00227 const NbOfEventsPair_T& lNbOfEventsPair = itNbOfEventsMap->second;
00228 stdair::Count_T lCurrentNbOfEvents = lNbOfEventsPair.first;
00229 const stdair::Count_T lExpectedTotalNbOfEvents = lNbOfEventsPair.second;
00230
00231 // Assess whether more events should be generated for that demand stream
00232 const bool stillHavingRequestsToBeGenerated = trademgenService.
00233     stillHavingRequestsToBeGenerated (lDemandStreamKey, lPPS,
00234     lDemandGenerationMethod);
00235
00242 if (lCurrentNbOfEvents == 1) {
00243     const stdair::ProgressStatus& lDemandStreamProgressStatus =
00244         lPPS.getSpecificGeneratorStatus();
00245     const stdair::Count_T& lNbOfRequests =
00246         lDemandStreamProgressStatus.getExpectedNb();
00252
00253     BOOST_CHECK_EQUAL (lNbOfRequests, lExpectedTotalNbOfEvents);
00254     BOOST_CHECK_MESSAGE (lNbOfRequests == lExpectedTotalNbOfEvents,
00255         "[" << lDemandStreamKey
00256         << "]" Total number of requests to be generated: "
00257         << lNbOfRequests << "). Expected value: "
00258         << lExpectedTotalNbOfEvents);
00259 }
00260
00261 // DEBUG
00262 STDAIR_LOG_DEBUG ("=> [" << lDemandStreamKey << "]"[" << lCurrentNbOfEvents
00263                  << "/" << lExpectedTotalNbOfEvents
00264                  << "] is now processed. "
00265                  << "Still generate events for that demand stream? "
00266                  << stillHavingRequestsToBeGenerated);
00267
00268 // If there are still events to be generated for that demand stream,
00269 // generate and add them to the event queue
00270 if (stillHavingRequestsToBeGenerated == true) {
00271     const stdair::BookingRequestPtr_T lNextRequest_ptr =
00272         trademgenService.generateNextRequest (lDemandStreamKey,
00273         lDemandGenerationMethod);
00274     assert (lNextRequest_ptr != NULL);
00275
00281     const stdair::Duration_T lDuration =
00282         lNextRequest_ptr->getRequestDateTime()
00283         - lPoppedRequest.getRequestDateTime();
00284     BOOST_REQUIRE_GT (lDuration.total_milliseconds(), 0);
00285     BOOST_REQUIRE_MESSAGE (lDuration.total_milliseconds() > 0,
00286         "[" << lDemandStreamKey
00287         << "] The date-time of the generated event ("
00288         << lNextRequest_ptr->getRequestDateTime()
00289         << ") is lower than the date-time "
00290         << "of the current event ("
00291         << lPoppedRequest.getRequestDateTime() << ")");
00292
00293 // DEBUG
00294 STDAIR_LOG_DEBUG ("[" << lDemandStreamKey << "]"[" << lCurrentNbOfEvents
00295                  << "/" << lExpectedTotalNbOfEvents
00296                  << "] Added request: '" << lNextRequest_ptr->describe()
00297                  << "'. Is queue done? "
00298                  << trademgenService.isQueueDone());
00299
00300 // Keep, within the dedicated map, the current counters of events
00301 // updated.
00302 ++lCurrentNbOfEvents;
00303 itNbOfEventsMap->second = NbOfEventsPair_T (lCurrentNbOfEvents,
00304     lExpectedTotalNbOfEvents);
00305 }
00306 // Iterate
00307 ++idx;
00308 }
00309 // Compensate for the last iteration
00310 --idx;
00311 //
00312 BOOST_CHECK_EQUAL (idx, lRefActualNbOfEvents);
00313 BOOST_CHECK_MESSAGE (idx == lRefActualNbOfEvents,

```

```

00314             "The total actual number of events is "
00315             << lRefActualNbOfEvents << ", but " << idx
00316             << " events have been generated");
00317
00320     trademgenService.reset();
00321
00322     // DEBUG
00323     STDAIR_LOG_DEBUG ("End of the simulation");
00324
00325     // Close the log file
00326     logOutputFile.close();
00327 }
00328
00329 // End the test suite
00330 BOOST_AUTO_TEST_SUITE_END()
00331
00332

```

25.17 test/trademgen/DemandGenerationTestSuite.hpp File Reference

```

#include <iosfwd>
#include <cppunit/extensions/HelperMacros.h>

```

Classes

- class [DemandGenerationTestSuite](#)

Functions

- [CPPUNIT_TEST_SUITE_REGISTRATION \(DemandGenerationTestSuite\)](#)

25.17.1 Function Documentation

25.17.1.1 CPPUNIT_TEST_SUITE_REGISTRATION (DemandGenerationTestSuite)

25.18 DemandGenerationTestSuite.hpp

```

00001 // STL
00002 #include <iosfwd>
00003 // CPPUNIT
00004 #include <cppunit/extensions/HelperMacros.h>
00005
00006 class DemandGenerationTestSuite : public CppUnit::TestFixture {
00007     CPPUNIT_TEST_SUITE (DemandGenerationTestSuite);
00008     CPPUNIT_TEST (simpleEventGeneration);
00009     // CPPUNIT_TEST (errorCase);
00010     CPPUNIT_TEST_SUITE_END ();
00011 public:
00012
00014     void simpleEventGeneration();
00015
00017     // void errorCase ();
00018
00020     DemandGenerationTestSuite ();
00021
00022 private:
00024     void simpleEventGenerationHelper();
00025
00026 protected:
00027     std::stringstream _describeKey;
00028 };
00029
00030 CPPUNIT_TEST_SUITE_REGISTRATION (DemandGenerationTestSuite);

```

25.19 test/trademgen/generateEvents.cpp File Reference

```

#include <cassert>

```

```

#include <string>
#include <map>
#include <iostream>
#include <sstream>
#include <test/trademgen/EventStream.hpp>
#include <test/trademgen/CategoricalAttribute.hpp>

```

Functions

- `int main (int argc, char *const argv[])`

25.19.1 Function Documentation

25.19.1.1 `int main (int argc, char *const argv[])`

Definition at line 12 of file [generateEvents.cpp](#).

25.20 generateEvents.cpp

```

00001 // STL
00002 #include <cassert>
00003 #include <string>
00004 #include <map>
00005 #include <iostream>
00006 #include <sstream>
00007 // TraDemGen
00008 #include <test/trademgen/EventStream.hpp>
00009 #include <test/trademgen/CategoricalAttribute.hpp>
00010
00011 // ////////////////////////////////// M A I N //////////////////////////////////
00012 int main (int argc, char* const argv[]) {
00013     // input: seed, rate
00014     unsigned long int seed = 2;
00015
00016     if (argc >= 2) {
00017         std::istringstream iStream (argv[1]);
00018         iStream >> seed;
00019     }
00020
00021     // create event stream
00022     TRADEMGEN::EventStream e (seed);
00023     e.setKey("hello");
00024     e.setRate(2.0);
00025
00026     // get rate
00027     // const double r = e.getRate();
00028     std::cout << "Seed: " << seed << std::endl << std::endl;
00029
00030     // create instances
00031     for (int i=0; i<10; i++) {
00032         e.generateNext();
00033     }
00034
00035     // display events
00036     e.displayAllEvents(std::cout);
00037
00038
00039     // //////////////////////////////////
00040     // attributes
00041     std::map<int, float> M;
00042     M[1] = 0.1;
00043     M[17] = 0.7;
00044     M[77] = 0.2;
00045     TRADEMGEN::CategoricalAttribute C (M);
00046
00047     return 0;
00048 }

```

25.21 trademgen/basic/BasConst.cpp File Reference

```
#include <stdair/basic/BasConst_General.hpp>
#include <trademgen/basic/BasConst_TRADEMGEN_Service.hpp>
#include <trademgen/basic/BasConst_DemandGeneration.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

Functions

- [stdair::BaseGenerator_T TRADEMGEN::DEFAULT_BASE_GENERATOR](#) (stdair::DEFAULT_RANDOM_SEED)
- [stdair::UniformGenerator_T TRADEMGEN::DEFAULT_UNIFORM_GENERATOR](#) (DEFAULT_BASE_GENERATOR, DEFAULT_UNIFORM_REAL_DISTRIBUTION)

Variables

- [const POSProbabilityMassFunction_T TRADEMGEN::DEFAULT_POS_PROBALILITY_MASS](#)
- [const stdair::FloatDuration_T TRADEMGEN::DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN](#) = -1
- [const FRAT5Pattern_T TRADEMGEN::DEFAULT_FRAT5_PATTERN](#) = DefaultMap::createFRAT5Pattern()
- [const double TRADEMGEN::DEFAULT_MAX_ADVANCE_PURCHASE](#) = 330.0
- [const stdair::UniformDistribution_T TRADEMGEN::DEFAULT_UNIFORM_REAL_DISTRIBUTION](#)

25.22 BasConst.cpp

```
00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // StdAir
00005 #include <stdair/basic/BasConst_General.hpp>
00006 // TraDemGen
00007 #include <trademgen/basic/BasConst_TRADEMGEN_Service.hpp>
00008 #include <trademgen/basic/BasConst_DemandGeneration.hpp>
00009
00010 namespace TRADEMGEN {
00011
00012     // const std::string DEFAULT_TRADEMGEN_SERVICE_NAME = "trademgen";
00013
00014     const POSProbabilityMassFunction_T DEFAULT_POS_PROBALILITY_MASS =
00015         DefaultMap::createPOSProbMass();
00016
00017     POSProbabilityMassFunction_T DefaultMap::createPOSProbMass() {
00018         POSProbabilityMassFunction_T oMap;
00019         // oMap["SIN"] = 0.44; oMap["HKG"] = 0.04; oMap["CGK"] = 0.04;
00020         // oMap["SYD"] = 0.04; oMap["BKK"] = 0.04; oMap["LHR"] = 0.03;
00021         // oMap["MEL"] = 0.03; oMap["KUL"] = 0.03; oMap["MNL"] = 0.03;
00022         // oMap["PVG"] = 0.03; oMap["PER"] = 0.02; oMap["BNE"] = 0.02;
00023         // oMap["NRT"] = 0.02; oMap["DPS"] = 0.02; oMap["SGN"] = 0.02;
00024         // oMap["PEN"] = 0.02; oMap["FRA"] = 0.02; oMap["PEK"] = 0.02;
00025         // oMap["HKT"] = 0.02; oMap["AKT"] = 0.02; oMap["SFO"] = 0.01;
00026         // oMap["ICN"] = 0.01; oMap["TPE"] = 0.01; oMap["row"] = 0.02;
00027         oMap["row"] = 1.0;
00028         return oMap;
00029     }
00030
00031     const stdair::FloatDuration_T DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN = -1;
00032
00033     const FRAT5Pattern_T DEFAULT_FRAT5_PATTERN = DefaultMap::createFRAT5Pattern();
00034
00035     FRAT5Pattern_T DefaultMap::createFRAT5Pattern() {
00036         FRAT5Pattern_T oMap;
00037         // oMap[1.10] = 0.0; oMap[1.40] = 0.80909; oMap[1.45] = 0.8303;
00038         // oMap[1.50] = 0.85152; oMap[1.55] = 0.87273; oMap[1.60] = 0.89394;
00039     }
```

```

00045 // oMap[1.70] = 0.90606; oMap[1.80] = 0.91818; oMap[2.00] = 0.9303;
00046 // oMap[2.30] = 0.94242; oMap[2.60] = 0.95152; oMap[3.00] = 0.96061;
00047 // oMap[3.30] = 0.96970; oMap[3.40] = 0.97879; oMap[3.44] = 0.98485;
00048 // oMap[3.47] = 0.99091; oMap[3.50] = 0.99697; oMap[3.500000001] = 1.0;
00049 // oMap[1.10] = -365; oMap[1.40] = -63; oMap[1.45] = -56;
00050 // oMap[1.50] = -49; oMap[1.55] = -42; oMap[1.60] = -35;
00051 // oMap[1.70] = -31; oMap[1.80] = -27; oMap[2.00] = -23;
00052 // oMap[2.30] = -19; oMap[2.60] = -16; oMap[3.00] = -13;
00053 // oMap[3.30] = -10; oMap[3.40] = -7; oMap[3.44] = -5;
00054 // oMap[3.47] = -3; oMap[3.50] = -1; oMap[3.500000001] = 0;
00055 // oMap[1.0] = -365; oMap[1.10] = -63; oMap[1.13] = -56;
00056 // oMap[1.17] = -49; oMap[1.22] = -42; oMap[1.28] = -35;
00057 // oMap[1.32] = -31; oMap[1.37] = -27; oMap[1.43] = -23;
00058 // oMap[1.51] = -19; oMap[1.60] = -16; oMap[1.70] = -13;
00059 // oMap[1.80] = -10; oMap[1.90] = -7; oMap[1.93] = -5;
00060 // oMap[1.96] = -3; oMap[2.00] = -1; oMap[2.000000001] = 0;
00061 // oMap[1.0] = -365; oMap[1.05] = -63; oMap[1.07] = -56;
00062 // oMap[1.09] = -49; oMap[1.11] = -42; oMap[1.14] = -35;
00063 // oMap[1.16] = -31; oMap[1.18] = -27; oMap[1.21] = -23;
00064 // oMap[1.24] = -19; oMap[1.27] = -16; oMap[1.30] = -13;
00065 // oMap[1.33] = -10; oMap[1.37] = -7; oMap[1.40] = -5;
00066 // oMap[1.45] = -3; oMap[1.50] = -1; oMap[1.500000001] = 0;
00067 oMap[1.10] = -365; oMap[1.40] = -63;
00068 oMap[1.50] = -49; oMap[1.60] = -35; oMap[2.00] = -23;
00069 oMap[2.60] = -16; oMap[3.30] = -10; oMap[3.44] = -5;
00070 oMap[3.50] = -1; oMap[3.500000001] = 0;
00071 return oMap;
00072 }
00073
00075 const double DEFAULT_MAX_ADVANCE_PURCHASE = 330.0;
00076
00078 stdair::BaseGenerator_T DEFAULT_BASE_GENERATOR (stdair::DEFAULT_RANDOM_SEED);
00079
00081 const stdair::UniformDistribution_T DEFAULT_UNIFORM_REAL_DISTRIBUTION;
00082
00084 stdair::UniformGenerator_T
00085 DEFAULT_UNIFORM_GENERATOR (DEFAULT_BASE_GENERATOR,
00086                             DEFAULT_UNIFORM_REAL_DISTRIBUTION);
00087
00088 }

```

25.23 trademgen/basic/BasConst_DemandGeneration.hpp File Reference

```

#include <string>
#include <stdair/stdair_maths_types.hpp>
#include <stdair/stdair_date_time_types.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>

```

Classes

- struct [TRADEMGEN::DefaultMap](#)

Namespaces

- namespace [TRADEMGEN](#)

Variables

- stdair::BaseGenerator_T [TRADEMGEN::DEFAULT_BASE_GENERATOR](#)
- stdair::UniformGenerator_T [TRADEMGEN::DEFAULT_UNIFORM_GENERATOR](#)

25.24 BasConst_DemandGeneration.hpp

```

00001 #ifndef __TRADEMGEN_BAS_BASCONST_DEMANDGENERATION_HPP
00002 #define __TRADEMGEN_BAS_BASCONST_DEMANDGENERATION_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section

```

```

00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_maths_types.hpp>
00011 #include <stdair/stdair_date_time_types.hpp>
00012 // TraDemGen
00013 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00014
00015 namespace TRADEMGEN {
00016
00018     extern const POSProbabilityMassFunction_T DEFAULT_POS_PROBALILITY_MASS;
00019
00021     extern const FRAT5Pattern_T DEFAULT_FRAT5_PATTERN;
00022
00024     struct DefaultMap {
00025         static POSProbabilityMassFunction_T createPOSProbMass();
00026         static FRAT5Pattern_T createFRAT5Pattern();
00027     };
00028
00030     extern const stdair::FloatDuration_T DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN
;
00031
00033     extern const double DEFAULT_MAX_ADVANCE_PURCHASE;
00034
00039     extern stdair::BaseGenerator_T DEFAULT_BASE_GENERATOR;
00040
00045     extern stdair::UniformGenerator_T DEFAULT_UNIFORM_GENERATOR;
00046
00048     extern const stdair::UniformDistribution_T DEFAULT_UNIFORM_REAL_DISTRIBUTION;
00049
00050 }
00051 #endif // __TRADEMGEN_BAS_BASCONST_DEMANDGENERATION_HPP

```

25.25 trademgen/basic/BasConst_TRADEMGEN_Service.hpp File Reference

```
#include <string>
```

Namespaces

- namespace [TRADEMGEN](#)

25.26 BasConst_TRADEMGEN_Service.hpp

```

00001 #ifndef __TRADEMGEN_BAS_BASCONST_TRADEMGEN_SERVICE_HPP
00002 #define __TRADEMGEN_BAS_BASCONST_TRADEMGEN_SERVICE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 #include <string>
00008
00009 namespace TRADEMGEN {
00010
00012     // extern const std::string DEFAULT_TRADEMGEN_SERVICE_NAME;
00013
00014 }
00015 #endif // __TRADEMGEN_BAS_BASCONST_TRADEMGEN_SERVICE_HPP

```

25.27 trademgen/basic/BasParserTypes.hpp File Reference

```

#include <string>
#include <boost/spirit/home/classic/core.hpp>
#include <boost/spirit/home/classic/utility/loops.hpp>
#include <boost/spirit/home/classic/utility/chset.hpp>
#include <boost/spirit/home/classic/utility/config.hpp>
#include <boost/spirit/home/classic/iterator/file_iterator.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef char [TRADEMGEN::char_t](#)
- typedef
boost::spirit::classic::file_iterator
< char_t > [TRADEMGEN::iterator_t](#)
- typedef
boost::spirit::classic::scanner
< iterator_t > [TRADEMGEN::scanner_t](#)
- typedef
boost::spirit::classic::rule
< scanner_t > [TRADEMGEN::rule_t](#)
- typedef
boost::spirit::classic::int_parser
< unsigned int, 10, 1, 1 > [TRADEMGEN::int1_p_t](#)
- typedef
boost::spirit::classic::uint_parser
< unsigned int, 10, 2, 2 > [TRADEMGEN::uint2_p_t](#)
- typedef
boost::spirit::classic::uint_parser
< unsigned int, 10, 1, 2 > [TRADEMGEN::uint1_2_p_t](#)
- typedef
boost::spirit::classic::uint_parser
< unsigned int, 10, 1, 3 > [TRADEMGEN::uint1_3_p_t](#)
- typedef
boost::spirit::classic::uint_parser
< unsigned int, 10, 4, 4 > [TRADEMGEN::uint4_p_t](#)
- typedef
boost::spirit::classic::uint_parser
< unsigned int, 10, 1, 4 > [TRADEMGEN::uint1_4_p_t](#)
- typedef
boost::spirit::classic::chset
< char_t > [TRADEMGEN::chset_t](#)
- typedef
boost::spirit::classic::impl::loop_traits
< chset_t, unsigned int,
unsigned int >::type [TRADEMGEN::repeat_p_t](#)
- typedef
boost::spirit::classic::bounded
< uint2_p_t, unsigned int > [TRADEMGEN::bounded2_p_t](#)
- typedef
boost::spirit::classic::bounded
< uint1_2_p_t, unsigned int > [TRADEMGEN::bounded1_2_p_t](#)
- typedef
boost::spirit::classic::bounded
< uint1_3_p_t, unsigned int > [TRADEMGEN::bounded1_3_p_t](#)
- typedef
boost::spirit::classic::bounded
< uint4_p_t, unsigned int > [TRADEMGEN::bounded4_p_t](#)
- typedef
boost::spirit::classic::bounded
< uint1_4_p_t, unsigned int > [TRADEMGEN::bounded1_4_p_t](#)

25.28 BasParserTypes.hpp

```

00001 #ifndef __TRADEMGEN_BAS_BASCOMPARSERTYPES_HPP
00002 #define __TRADEMGEN_BAS_BASCOMPARSERTYPES_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // Boost
00010 // #define BOOST_SPIRIT_DEBUG
00011 #include <boost/spirit/home/classic/core.hpp>
00012 // #include <boost/spirit/home/classic/attribute.hpp>
00013 // #include <boost/spirit/home/classic/utility/functor_parser.hpp>
00014 #include <boost/spirit/home/classic/utility/loops.hpp>
00015 #include <boost/spirit/home/classic/utility/chset.hpp>
00016 #include <boost/spirit/home/classic/utility/confix.hpp>
00017 #include <boost/spirit/home/classic/iterator/file_iterator.hpp>
00018 // #include <boost/spirit/home/classic/actor/push_back_actor.hpp>
00019 // #include <boost/spirit/home/classic/actor/assign_actor.hpp>
00020
00021 namespace TRADEMGEN {
00022
00023 // //////////////////////////////////////
00024 //
00025 // Definition of Basic Types
00026 //
00027 // //////////////////////////////////////
00028 // For a file, the parsing unit is the character (char). For a string,
00029 // it is a "char const *".
00030 // typedef char const* iterator_t;
00031 typedef char char_t;
00032
00033 // The types of iterator, scanner and rule are then derived from
00034 // the parsing unit.
00035 typedef boost::spirit::classic::file_iterator<char_t> iterator_t;
00036 typedef boost::spirit::classic::scanner<iterator_t> scanner_t;
00037 typedef boost::spirit::classic::rule<scanner_t> rule_t;
00038
00039 // //////////////////////////////////////
00040 //
00041 // Parser related types
00042 //
00043 // //////////////////////////////////////
00044 typedef boost::spirit::classic::int_parser<unsigned int, 10, 1, 1> int1_p_t;
00045
00046
00047 typedef boost::spirit::classic::uint_parser<unsigned int, 10, 2, 2> uint2_p_t
00048 ;
00049
00050 typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 2>
00051 uint1_2_p_t;
00052
00053 typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 3>
00054 uint1_3_p_t;
00055
00056 typedef boost::spirit::classic::uint_parser<unsigned int, 10, 4, 4> uint4_p_t
00057 ;
00058
00059 typedef boost::spirit::classic::uint_parser<unsigned int, 10, 1, 4>
00060 uint1_4_p_t;
00061
00062 typedef boost::spirit::classic::chset<char_t> chset_t;
00063
00064
00065 typedef boost::spirit::classic::impl::loop_traits<chset_t,
00066 unsigned int,
00067 unsigned int>::type repeat_p_t;
00068
00069
00070 typedef boost::spirit::classic::bounded<uint2_p_t, unsigned int> bounded2_p_t
00071 ;
00072
00073 typedef boost::spirit::classic::bounded<uint1_2_p_t, unsigned int>
00074 bounded1_2_p_t;
00075 typedef boost::spirit::classic::bounded<uint1_3_p_t, unsigned int>
00076 bounded1_3_p_t;
00077 typedef boost::spirit::classic::bounded<uint4_p_t, unsigned int> bounded4_p_t
00078 ;
00079
00080 typedef boost::spirit::classic::bounded<uint1_4_p_t, unsigned int>
00081 bounded1_4_p_t;
00082 }
00083 #endif // __TRADEMGEN_BAS_BASCOMPARSERTYPES_HPP

```

25.29 trademgen/basic/CategoricalAttribute.hpp File Reference

```
#include <map>
#include <iosfwd>
#include <stdair/STDAIR_Types.hpp>
#include <stdair/basic/DictionaryManager.hpp>
```

Classes

- struct [stdair::CategoricalAttribute< T >](#)

Class modeling the distribution of values that can be taken by a categorical attribute.

Namespaces

- namespace [stdair](#)

Forward declarations.

25.30 CategoricalAttribute.hpp

```
00001 #ifndef __STDAIR_BAS_CATEGORICALATTRIBUTE_HPP
00002 #define __STDAIR_BAS_CATEGORICALATTRIBUTE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <map>
00009 #include <iosfwd>
00010 // STDAIR
00011 #include <stdair/STDAIR_Types.hpp>
00012 #include <stdair/basic/DictionaryManager.hpp>
00013
00014 namespace stdair {
00015
00020     template <typename T>
00021     struct CategoricalAttribute {
00022
00023     public:
00024         // ////////////////////////////////// Type definitions //////////////////////////////////
00028         typedef std::map<T, DictionaryKey_T> ProbabilityMassFunction_T;
00029
00033         typedef std::map<DictionaryKey_T, T> InverseCumulativeDistribution_T;
00034
00035     private:
00037         // ////////////////////////////////// Getters //////////////////////////////////
00041         const ProbabilityMassFunction_T& getProbabilityMassFunction() const {
00042             return _probabilityMassFunction;
00043         }
00044
00048         const InverseCumulativeDistribution_T& getInverseCumulativeDistribution()
00049         const {
00050             return _inverseCumulativeDistribution;
00051         }
00052         // ////////////////////////////////// Setters //////////////////////////////////
00056         void setProbabilityMassFunction (const ProbabilityMassFunction_T&
00057 iProbabilityMassFunction) {
00058             _probabilityMassFunction = iProbabilityMassFunction;
00059             determineInverseCumulativeDistributionFromProbabilityMassFunction();
00060         }
00061
00062     public:
00063         // ////////////////////////////////// Business Methods //////////////////////////////////
00067         const T& getValue (const Probability_T& iCumulativeProbability) const {
00068
00069             const DictionaryKey_T& lKey =
00070                 DictionaryManager::valueToKey (iCumulativeProbability);
00071
00072             InverseCumulativeDistribution_T::const_iterator itT =
00073                 _inverseCumulativeDistribution.find (lKey);
```

```

00074
00075     if (itT == _inverseCumulativeDistribution.end()) {
00076         std::ostringstream ostr;
00077         ostr << "The following cumulative probability is out of range: "
00078             << iCumulativeProbability << displayInverseCumulativeDistribution(
);
00079         throw IndexOutOfRangeException (ostr.str());
00080     }
00081
00082     return itT->second;
00083 }
00084
00085
00086 public:
00087 // //////////// Display Support Methods ////////////
00091 const std::string displayProbabilityMassFunction() const {
00092     std::ostringstream ostr;
00093     unsigned int idx = 0;
00094
00095     for (typename ProbabilityMassFunction_T::const_iterator it =
00096         _probabilityMassFunction.begin();
00097         it != _probabilityMassFunction.end(); ++it, ++idx) {
00098         if (idx != 0) {
00099             ostr << ", ";
00100         }
00101         ostr << it->first << ":"
00102             << DictionaryManager::keyToValue (it->second);
00103     }
00104
00105     return ostr.str();
00106 }
00107
00111 const std::string displayInverseCumulativeDistribution() const {
00112     std::ostringstream ostr;
00113
00114     for (typename InverseCumulativeDistribution_T::const_iterator it =
00115         _inverseCumulativeDistribution.begin();
00116         it != _inverseCumulativeDistribution.end(); ++it) {
00117         ostr << "cumulative prob: " << DictionaryManager::keyToValue (it->first
);
00118             << " value: " << it->second << std::endl;
00119     }
00120
00121     return ostr.str();
00122 }
00123
00124 public:
00125 // //////////// Constructors and destructors ////////////
00129 CategoricalAttribute (const ProbabilityMassFunction_T&
iProbabilityMassFunction)
00130 : _probabilityMassFunction (iProbabilityMassFunction) {
00131     determineInverseCumulativeDistributionFromProbabilityMassFunction();
00132 }
00133
00137 CategoricalAttribute() { }
00138
00142 CategoricalAttribute (const CategoricalAttribute& iCategoricalAttribute)
00143 : _probabilityMassFunction (iCategoricalAttribute.
_probabilityMassFunction) {
00144     determineInverseCumulativeDistributionFromProbabilityMassFunction();
00145 }
00146
00150 virtual ~CategoricalAttribute() { }
00151
00152
00157 void determineInverseCumulativeDistributionFromProbabilityMassFunction() {
00158
00159     Probability_T cumulative_probability_so_far = 0.0;
00160     for (typename ProbabilityMassFunction_T::const_iterator
00161         itProbabilityMassFunction = _probabilityMassFunction.begin();
00162         itProbabilityMassFunction != _probabilityMassFunction.end();
00163         ++itProbabilityMassFunction) {
00164
00165         Probability_T attribute_probability_mass =
00166             DictionaryManager::keyToValue (itProbabilityMassFunction->second);
00167
00168         if (attribute_probability_mass > 0) {
00169             T attribute_value = itProbabilityMassFunction->first;
00170             cumulative_probability_so_far += attribute_probability_mass;
00171
00172             const DictionaryKey_T& lKey =
00173                 DictionaryManager::valueToKey (cumulative_probability_so_far);
00174
00175             //_inverseCumulativeDistribution[lKey] = attribute_value;
00176             _inverseCumulativeDistribution.
00177                 insert (typename InverseCumulativeDistribution_T::
00178                     value_type (lKey, attribute_value));

```

```

00179     }
00180     }
00181     }
00182
00183 private:
00184     // ////////// Attributes //////////
00188     ProbabilityMassFunction_T _probabilityMassFunction;
00189
00193     InverseCumulativeDistribution_T _inverseCumulativeDistribution;
00194 };
00195 }
00196 #endif // __STDAIR_BAS_CATEGORICALATTRIBUTE_HPP

```

25.31 trademgen/basic/CategoricalAttributeLite.hpp File Reference

```

#include <cassert>
#include <sstream>
#include <string>
#include <vector>
#include <map>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/TRADEMGEN_Exceptions.hpp>
#include <trademgen/basic/DictionaryManager.hpp>

```

Classes

- struct [TRADEMGEN::CategoricalAttributeLite< T >](#)
Class modeling the distribution of values that can be taken by a categorical attribute.

Namespaces

- namespace [TRADEMGEN](#)

25.32 CategoricalAttributeLite.hpp

```

00001 #ifndef __TRADEMGEN_BAS_CATEGORICALATTRIBUTE_LITE_HPP
00002 #define __TRADEMGEN_BAS_CATEGORICALATTRIBUTE_LITE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <cassert>
00009 #include <sstream>
00010 #include <string>
00011 #include <vector>
00012 #include <map>
00013 // StdAir
00014 #include <stdair/stdair_basic_types.hpp>
00015 #include <stdair/service/Logger.hpp>
00016 // TraDemGen
00017 #include <trademgen/TRADEMGEN_Exceptions.hpp>
00018 #include <trademgen/basic/DictionaryManager.hpp>
00019
00020 namespace TRADEMGEN {
00021
00026     template <typename T>
00027     struct CategoricalAttributeLite {
00028     public:
00029         // ////////// Type definitions //////////
00033         typedef std::map<T, stdair::Probability_T> ProbabilityMassFunction_T;
00034
00035
00036     public:
00037         // ////////// Business Methods //////////
00041         const T& getValue (const stdair::Probability_T& iCumulativeProbability)
00041         const {

```

```

00042     const DictionaryKey_T& lKey =
00043         DictionaryManager::valueToKey (iCumulativeProbability);
00044
00045     for (unsigned int idx = 0; idx < _size; ++idx) {
00046         if (_cumulativeDistribution.at(idx) >= lKey) {
00047             const T& oValue = _valueArray.at(idx);
00048             return oValue;
00049         }
00050     }
00051
00052     std::ostringstream ostr;
00053     ostr << "The following cumulative probability is out of range: "
00054         << iCumulativeProbability << displayProbabilityMass();
00055     throw IndexOutOfRangeException (ostr.str());
00056 }
00057
00061 bool checkValue (const T& iValue) const {
00062     for (unsigned int idx = 0; idx < _size; ++idx) {
00063         if (_valueArray.at(idx) == iValue) {
00064             return true;
00065         }
00066     }
00067     return false;
00068 }
00069
00070 public:
00071 // //////////// Display Support Methods ////////////
00072 const std::string displayProbabilityMass() const {
00073     std::ostringstream ostr;
00074
00075     for (unsigned int idx = 0; idx < _size; ++idx) {
00076         if (idx != 0) {
00077             ostr << ", ";
00078         }
00079         ostr << _valueArray.at(idx) << ":"
00080             << DictionaryManager::keyToValue (_cumulativeDistribution[idx]);
00081     }
00082     return ostr.str();
00083 }
00084
00085 public:
00086 // //////////// Constructors and destructors ////////////
00087 CategoricalAttributeLite (const ProbabilityMassFunction_T& iValueMap)
00088 : _size (iValueMap.size()) {
00089     init (iValueMap);
00090 }
00091
00092 CategoricalAttributeLite() : _size(1) {
00093 }
00094
00095 CategoricalAttributeLite (const CategoricalAttributeLite& iCAL)
00096 : _size (iCAL._size),
00097   _cumulativeDistribution (iCAL._cumulativeDistribution),
00098   _valueArray (iCAL._valueArray) {
00099 }
00100
00101 CategoricalAttributeLite& operator= (const CategoricalAttributeLite& iCAL)
00102 {
00103     _size = iCAL._size;
00104     _cumulativeDistribution = iCAL._cumulativeDistribution;
00105     _valueArray = iCAL._valueArray;
00106     return *this;
00107 }
00108
00109 virtual ~CategoricalAttributeLite() {
00110 }
00111
00112 private:
00113 void init (const ProbabilityMassFunction_T& iValueMap) {
00114
00115     const unsigned int lSize = iValueMap.size();
00116     _cumulativeDistribution.reserve (lSize);
00117     _valueArray.reserve (lSize);
00118
00119     std::dair::Probability_T cumulative_probability_so_far = 0.0;
00120
00121     // Browse the map to retrieve the values and to build the
00122     // cumulative probabilities.
00123     for (typename ProbabilityMassFunction_T::const_iterator
00124         itProbabilityMassFunction = iValueMap.begin();
00125         itProbabilityMassFunction != iValueMap.end();
00126         ++itProbabilityMassFunction) {
00127
00128         std::dair::Probability_T attribute_probability_mass =

```

```

00152         itProbabilityMassFunction->second;
00153
00154         if (attribute_probability_mass > 0) {
00155             const T& attribute_value = itProbabilityMassFunction->first;
00156             cumulative_probability_so_far += attribute_probability_mass;
00157
00158             const DictionaryKey_T& lKey =
00159                 DictionaryManager::valueToKey (cumulative_probability_so_far);
00160
00161             // Build the two arrays.
00162             _cumulativeDistribution.push_back (lKey);
00163             _valueArray.push_back (attribute_value);
00164         }
00165     }
00166 }
00167
00168 private:
00169     // //////////// Attributes ////////////
00170     unsigned int _size;
00171
00172     std::vector<DictionaryKey_T> _cumulativeDistribution;
00173
00174     std::vector<T> _valueArray;
00175 };
00176
00177 #endif // __TRADEMGEN_BAS_CATEGORICALATTRIBUTE_LITE_HPP

```

25.33 trademgen/basic/ContinuousAttribute.hpp File Reference

```

#include <string>
#include <map>
#include <stdair/stdair_date_time_types.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/basic/DictionaryManager.hpp>

```

Classes

- struct [TRADEMGEN::ContinuousAttribute< T >](#)

Namespaces

- namespace [TRADEMGEN](#)

25.34 ContinuousAttribute.hpp

```

00001 #ifndef __TRADEMGEN_BAS_CONTINUOUSATTRIBUTE_HPP
00002 #define __TRADEMGEN_BAS_CONTINUOUSATTRIBUTE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 #include <map>
00010 // StdAir
00011 #include <stdair/stdair_date_time_types.hpp>
00012 #include <stdair/service/Logger.hpp>
00013 // TraDemGen
00014 #include <trademgen/basic/DictionaryManager.hpp>
00015
00016 namespace TRADEMGEN {
00017
00020     template <class T>
00021     struct ContinuousAttribute {
00022     public:
00023
00024         // //////////// Type definitions ////////////
00026         typedef std::multimap<T, DictionaryKey_T> ContinuousDistribution_T;
00027         typedef std::multimap<DictionaryKey_T, T> ContinuousInverseDistribution_T;
00028
00029     private:

```

```

00030 // //////////// Getters ////////////
00032 const ContinuousDistribution_T& getCumulativeDistribution() const {
00033     return _cumulativeDistribution;
00034 }
00035
00037 const ContinuousInverseDistribution_T& getInverseCumulativeDistribution ()
const {
00038     return _inverseCumulativeDistribution;
00039 }
00040
00041 private:
00042 // //////////// Setters ////////////
00044 void setCumulativeDistribution (const ContinuousDistribution_T&
iCumulativeDistribution) {
00045     _cumulativeDistribution = iCumulativeDistribution;
00046     determineInverseCumulativeDistributionFromCumulativeDistribution();
00047 }
00048
00049 public:
00050 // //////////// Business Methods ////////////
00052 const T getValue (const stdair::Probability_T& iCumulativeProbability)
const{
00053     const DictionaryKey_T lKey =
DictionaryManager::valueToKey (iCumulativeProbability);
00054     typename ContinuousInverseDistribution_T::const_iterator it =
_inverseCumulativeDistribution.lower_bound (lKey);
00055
00056     stdair::Probability_T cumulativeProbabilityNextPoint =
DictionaryManager::keyToValue (it->first);
00057     T valueNextPoint = it->second;
00058
00059     if (it == _inverseCumulativeDistribution.begin()) {
00060         STDAIR_LOG_DEBUG ("Last element");
00061         return valueNextPoint;
00062     }
00063     --it;
00064
00065     stdair::Probability_T cumulativeProbabilityPreviousPoint =
DictionaryManager::keyToValue (it->first);
00066     T valuePreviousPoint = it->second;
00067     if (cumulativeProbabilityNextPoint == cumulativeProbabilityPreviousPoint)
{
00068         return valuePreviousPoint;
00069     }
00070
00071     return valuePreviousPoint + (valueNextPoint - valuePreviousPoint)
* (iCumulativeProbability - cumulativeProbabilityPreviousPoint)
/ (cumulativeProbabilityNextPoint - cumulativeProbabilityPreviousPoint)
;
00072 }
00073
00074 public:
00075 // //////////// Display Support Methods ////////////
00076 const std::string displayCumulativeDistribution() const {
00077     std::ostringstream oStr;
00078     unsigned int idx = 0;
00079     for (typename ContinuousDistribution_T::const_iterator it =
_cumulativeDistribution.begin();
00080          it != _cumulativeDistribution.end(); ++it, ++idx) {
00081         if (idx != 0) {
00082             oStr << ", ";
00083         }
00084         oStr << it->first << ":"
<< DictionaryManager::keyToValue (it->second);
00085     }
00086     return oStr.str();
00087 }
00088
00089 const std::string displayInverseCumulativeDistribution() const {
00090     std::ostringstream oStr;
00091     for (typename ContinuousInverseDistribution_T::const_iterator it =
_inverseCumulativeDistribution.begin();
00092          it != _inverseCumulativeDistribution.end(); ++it) {
00093         oStr << "cumulative prob: " << DictionaryManager::keyToValue (it->first)
<< " value: " << it->second << std::endl;
00094     }
00095     return oStr.str();
00096 }
00097
00098 public:
00099 // //////////// Constructors and destructors ////////////
00100 ContinuousAttribute () { }
00101
00102 ContinuousAttribute (const ContinuousDistribution_T&
iCumulativeDistribution)
: _cumulativeDistribution (iCumulativeDistribution) {
00103

```

```

00118     determineInverseCumulativeDistributionFromCumulativeDistribution();
00119 }
00120
00122 ContinuousAttribute (const ContinuousAttribute& iContinuousAttribute)
00123 : _cumulativeDistribution (iContinuousAttribute._cumulativeDistribution),
00124   _inverseCumulativeDistribution (iContinuousAttribute.
_inverseCumulativeDistribution) {
00125 }
00126
00128 virtual ~ContinuousAttribute () { }
00129
00132 void determineInverseCumulativeDistributionFromCumulativeDistribution () {
00133     for (typename ContinuousDistribution_T::iterator itCumulativeDistribution
=
00134         _cumulativeDistribution.begin();
00135         itCumulativeDistribution != _cumulativeDistribution.end();
00136         ++itCumulativeDistribution) {
00137         _inverseCumulativeDistribution.
00138             insert (typename ContinuousInverseDistribution_T::
00139                 value_type (itCumulativeDistribution->second,
00140                     itCumulativeDistribution->first));
00141     }
00142 }
00143
00144 private:
00145     // ////////// Attributes //////////
00146
00148     ContinuousDistribution_T _cumulativeDistribution;
00149
00151     ContinuousInverseDistribution_T _inverseCumulativeDistribution;
00152 };
00153
00154 }
00155 #endif // __STDAIR_BAS_CONTINUOUSATTRIBUTE_HPP

```

25.35 trademgen/basic/ContinuousAttributeLite.hpp File Reference

```

#include <cassert>
#include <iosfwd>
#include <string>
#include <vector>
#include <map>
#include <stdair/stdair_basic_types.hpp>
#include <trademgen/TRADEMGEX_Exceptions.hpp>
#include <trademgen/basic/DictionaryManager.hpp>

```

Classes

- struct [TRADEMGEX::ContinuousAttributeLite< T >](#)
Class modeling the distribution of values that can be taken by a continuous attribute.

Namespaces

- namespace [TRADEMGEX](#)

25.36 ContinuousAttributeLite.hpp

```

00001 #ifndef __TRADEMGEX_BAS_CONTINUOUSATTRIBUTE_LITE_HPP
00002 #define __TRADEMGEX_BAS_CONTINUOUSATTRIBUTE_LITE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <cassert>
00009 #include <iosfwd>
00010 #include <string>
00011 #include <vector>
00012 #include <map>

```



```

00013 // StdAir
00014 #include <stdair/stdair_basic_types.hpp>
00015 // TraDemGen
00016 #include <trademgen/TRADEMGEN_Exceptions.hpp>
00017 #include <trademgen/basic/DictionaryManager.hpp>
00018
00019 namespace TRADEMGEN {
00020
00021     template <typename T>
00022     struct ContinuousAttributeLite {
00023     public:
00024         // ////////////////////////////////// Type definitions //////////////////////////////////
00025         typedef std::map<T, stdair::Probability_T> ContinuousDistribution_T;
00026
00027     public:
00028         // ////////////////////////////////// Business Methods //////////////////////////////////
00029         const T getValue(const stdair::Probability_T& iCumulativeProbability) const
00030         {
00031             const DictionaryKey_T& lKey =
00032                 DictionaryManager::valueToKey (iCumulativeProbability);
00033
00034             // Find the first cumulative probability value greater or equal to lKey.
00035             unsigned int idx = 0;
00036             for (; idx < _size; ++idx) {
00037                 if (_cumulativeDistribution.at(idx) > lKey) {
00038                     break;
00039                 }
00040             }
00041
00042             if (idx == 0) {
00043                 return _valueArray.at(idx);
00044             }
00045             if (idx == _size) {
00046                 return _valueArray.at(idx-1);
00047             }
00048
00049             //
00050             const stdair::Probability_T& lCumulativeCurrentPoint =
00051                 DictionaryManager::keyToValue (_cumulativeDistribution.at(idx));
00052             const T& lValueCurrentPoint = _valueArray.at(idx);
00053
00054             //
00055             const stdair::Probability_T& lCumulativePreviousPoint =
00056                 DictionaryManager::keyToValue (_cumulativeDistribution.at(idx-1));
00057             const T& lValuePreviousPoint = _valueArray.at(idx-1);
00058
00059             if (lCumulativePreviousPoint == lCumulativeCurrentPoint) {
00060                 return lValuePreviousPoint;
00061             }
00062
00063             T oValue= lValuePreviousPoint + (lValueCurrentPoint - lValuePreviousPoint
00064
00065                 * (iCumulativeProbability - lCumulativePreviousPoint)
00066                 / (lCumulativeCurrentPoint - lCumulativePreviousPoint);
00067
00068             return oValue;
00069         }
00070
00071         const double getDerivativeValue(const T iKey) const{
00072
00073             // Find the first key value greater or equal to iKey.
00074             unsigned int idx = 0;
00075             for (; idx < _size; ++idx) {
00076                 if (_valueArray.at(idx) > iKey) {
00077                     break;
00078                 }
00079             }
00080
00081             assert (idx != 0);
00082             assert (idx != _size);
00083
00084             //
00085             const stdair::Probability_T& lCumulativeCurrentPoint =
00086                 DictionaryManager::keyToValue (_cumulativeDistribution.at(idx));
00087             const T& lValueCurrentPoint = _valueArray.at(idx);
00088
00089             //
00090             const stdair::Probability_T& lCumulativePreviousPoint =
00091                 DictionaryManager::keyToValue (_cumulativeDistribution.at(idx-1));
00092             const T& lValuePreviousPoint = _valueArray.at(idx-1);
00093
00094             assert (lValueCurrentPoint != lValuePreviousPoint);
00095
00096             const double oValue= (lCumulativeCurrentPoint - lCumulativePreviousPoint)
00097                 / (lValueCurrentPoint - lValuePreviousPoint);
00098
00099             return oValue;
00100

```

```

00111     }
00112
00116     const T getUpperBound (const T iKey) const {
00117         // Find the first key value greater or equal to iKey.
00118         unsigned int idx = 0;
00119         for (; idx < _size; ++idx) {
00120             if (_valueArray.at(idx) > iKey) {
00121                 break;
00122             }
00123         }
00124         assert (idx != 0);
00125         assert (idx != _size);
00126         return _valueArray.at (idx);
00127     }
00128
00129 public:
00130 // ////////////////////////////////// Display Support Methods //////////////////////////////////
00131 const std::string displayCumulativeDistribution() const {
00132     std::ostringstream oStr;
00133
00134     for (unsigned int idx = 0; idx < _size; ++idx) {
00135         if (idx != 0) {
00136             oStr << ", ";
00137         }
00138
00139         const stdair::Probability_T& lProbability =
00140             DictionaryManager::keyToValue (_cumulativeDistribution.at(idx));
00141
00142         oStr << _valueArray.at(idx) << ":" << lProbability;
00143     }
00144     return oStr.str();
00145 }
00146
00147 public:
00148 // ////////////////////////////////// Constructors and destructors //////////////////////////////////
00149 ContinuousAttributeLite (const ContinuousDistribution_T& iValueMap)
00150 : _size (iValueMap.size()) {
00151     init (iValueMap);
00152 }
00153
00154 ContinuousAttributeLite (const ContinuousAttributeLite& iCAL)
00155 : _size (iCAL._size),
00156   _cumulativeDistribution (iCAL._cumulativeDistribution),
00157   _valueArray (iCAL._valueArray) {
00158 }
00159
00160 ContinuousAttributeLite& operator= (const ContinuousAttributeLite& iCAL) {
00161     _size = iCAL._size;
00162     _cumulativeDistribution = iCAL._cumulativeDistribution;
00163     _valueArray = iCAL._valueArray;
00164     return *this;
00165 }
00166
00167 virtual ~ContinuousAttributeLite() {
00168 }
00169
00170 private:
00171 ContinuousAttributeLite() : _size(1) {
00172 }
00173
00174 void init (const ContinuousDistribution_T& iValueMap) {
00175     //
00176     const unsigned int lSize = iValueMap.size();
00177     _cumulativeDistribution.reserve (lSize);
00178     _valueArray.reserve (lSize);
00179
00180     // Browse the map to retrieve the values and cumulative probabilities.
00181     for (typename ContinuousDistribution_T::const_iterator it =
00182         iValueMap.begin(); it != iValueMap.end(); ++it) {
00183
00184         const T& attributeValue = it->first;
00185         const DictionaryKey_T& lKey = DictionaryManager::valueToKey (it->second
00186     );
00187
00188         // Build the two arrays.
00189         _cumulativeDistribution.push_back (lKey);
00190         _valueArray.push_back (attributeValue);
00191     }
00192 }
00193
00194 private:
00195 // ////////////////////////////////// Attributes //////////////////////////////////
00196 unsigned int _size;
00197

```

```

00228     std::vector<DictionaryKey_T> _cumulativeDistribution;
00229
00233     std::vector<T> _valueArray;
00234 };
00235
00236 }
00237 #endif // __TRADEMGEN_BAS_CONTINUOUSATTRIBUTELITE_HPP

```

25.37 trademgen/basic/DemandCharacteristics.cpp File Reference

```

#include <cassert>
#include <sstream>
#include <stdair/stdair_basic_types.hpp>
#include <trademgen/basic/BasConst_DemandGeneration.hpp>
#include <trademgen/basic/DemandCharacteristics.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.38 DemandCharacteristics.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 // TraDemGen
00010 #include <trademgen/basic/BasConst_DemandGeneration.hpp>
00011 #include <trademgen/basic/DemandCharacteristics.hpp>
00012
00013 namespace TRADEMGEN {
00014
00015     // //////////////////////////////////////
00016     DemandCharacteristics::DemandCharacteristics()
00017     : _arrivalPattern (ArrivalPatternCumulativeDistribution_T()),
00018       _posProbabilityMass (POSPProbabilityMassFunction_T()),
00019       _channelProbabilityMass (ChannelProbabilityMassFunction_T()),
00020       _tripTypeProbabilityMass (TripTypeProbabilityMassFunction_T()),
00021       _stayDurationProbabilityMass (StayDurationProbabilityMassFunction_T()),
00022       _frequentFlyerProbabilityMass (FrequentFlyerProbabilityMassFunction_T()),
00023       _preferredDepartureTimeCumulativeDistribution (
00024         PreferredDepartureTimeContinuousDistribution_T()),
00024       _minWTP (stdair::WTP_T()), _frat5Pattern (DEFAULT_FRAT5_PATTERN),
00025       _valueOfTimeCumulativeDistribution (ValueOfTimeContinuousDistribution_T())
00026     {}
00027
00028     // //////////////////////////////////////
00029     DemandCharacteristics::
00030     DemandCharacteristics (const DemandCharacteristics& iDC)
00031     : _arrivalPattern (iDC._arrivalPattern),
00032       _posProbabilityMass (iDC._posProbabilityMass),
00033       _channelProbabilityMass (iDC._channelProbabilityMass),
00034       _tripTypeProbabilityMass (iDC._tripTypeProbabilityMass),
00035       _stayDurationProbabilityMass (iDC._stayDurationProbabilityMass),
00036       _frequentFlyerProbabilityMass (iDC._frequentFlyerProbabilityMass),
00037       _preferredDepartureTimeCumulativeDistribution (iDC.
00038         _preferredDepartureTimeCumulativeDistribution),
00038       _minWTP (iDC._minWTP), _frat5Pattern (iDC._frat5Pattern),
00039       _valueOfTimeCumulativeDistribution (iDC.
00040         _valueOfTimeCumulativeDistribution) {}
00041
00042     // //////////////////////////////////////
00043     DemandCharacteristics::
00044     DemandCharacteristics (const ArrivalPatternCumulativeDistribution_T&
00045       iArrivalPattern,
00045       const POSProbabilityMassFunction_T& iPOSProbMass,
00046       const ChannelProbabilityMassFunction_T&
00047       iChannelProbMass,

```

```

00047         const TripTypeProbabilityMassFunction_T&
00048         iTripTypeProbMass,
00049         const StayDurationProbabilityMassFunction_T&
00050         iStayDurationProbMass,
00051         const FrequentFlyerProbabilityMassFunction_T&
00052         iFrequentFlyerProbMass,
00053         const PreferredDepartureTimeContinuousDistribution_T&
00054         iPreferredDepartureTimeContinuousDistribution,
00055         const stdair::WTP_T& iMinWTP,
00056         const ValueOfTimeContinuousDistribution_T&
00057         iValueOfTimeContinuousDistribution)
00058     : _arrivalPattern (iArrivalPattern),
00059       _posProbabilityMass (iPOSProbMass),
00060       _channelProbabilityMass (iChannelProbMass),
00061       _tripTypeProbabilityMass (iTripTypeProbMass),
00062       _stayDurationProbabilityMass (iStayDurationProbMass),
00063       _frequentFlyerProbabilityMass (iFrequentFlyerProbMass),
00064       _preferredDepartureTimeCumulativeDistribution (
00065         iPreferredDepartureTimeContinuousDistribution),
00066       _minWTP (iMinWTP), _frat5Pattern (DEFAULT_FRAT5_PATTERN),
00067       _valueOfTimeCumulativeDistribution (iValueOfTimeContinuousDistribution) {
00068     }
00069
00070     // //////////////////////////////////////
00071     DemandCharacteristics::~DemandCharacteristics() {
00072     }
00073
00074     // //////////////////////////////////////
00075     const stdair::AirportCode_T& DemandCharacteristics::
00076     getPOSValue (const stdair::Probability_T& iCumulativeProbability) const {
00077         return _posProbabilityMass.getValue (iCumulativeProbability);
00078     }
00079
00080     // //////////////////////////////////////
00081     bool DemandCharacteristics::
00082     checkPOSValue (const stdair::AirportCode_T& iPOS) const {
00083         return _posProbabilityMass.checkValue (iPOS);
00084     }
00085
00086     // //////////////////////////////////////
00087     const std::string DemandCharacteristics::describe() const {
00088         std::ostringstream oStr;
00089
00090         //
00091         oStr << "***** Demand characteristics *****"
00092         << std::endl;
00093         oStr << "Arrival pattern (days from departure, proportion): ";
00094         oStr << _arrivalPattern.displayCumulativeDistribution() << std::endl;
00095         oStr << "POS probability mass (POS, propotion): ";
00096         oStr << _posProbabilityMass.displayProbabilityMass()
00097         << std::endl;
00098         oStr << "Channel probability mass (channel, propotion): ";
00099         oStr << _channelProbabilityMass.displayProbabilityMass()
00100         << std::endl;
00101         oStr << "Trip type probability mass (trip type, propotion): ";
00102         oStr << _tripTypeProbabilityMass.displayProbabilityMass()
00103         << std::endl;
00104         oStr << "Stay duration probability mass (number of days, propotion): ";
00105         oStr << _stayDurationProbabilityMass.displayProbabilityMass()
00106         << std::endl;
00107         oStr << "Frequent flyer probability mass (frequent flyer, propotion): ";
00108         oStr << _frequentFlyerProbabilityMass.displayProbabilityMass()
00109         << std::endl;
00110         oStr << "Preferred departure time cumulative distribution (time,
00111         proportion: ";
00112         oStr << _preferredDepartureTimeCumulativeDistribution.
00113         displayCumulativeDistribution() << std::endl;
00114         oStr << "min WTP: " << _minWTP << std::endl;
00115         oStr << "Value of time cumulative distribution (value of time, proportion:
00116         ";
00117         oStr << _valueOfTimeCumulativeDistribution.displayCumulativeDistribution()
00118         << std::endl;
00119
00120         return oStr.str();
00121     }
00122 }
00123
00124 }
```

25.39 trademgen/basic/DemandCharacteristics.hpp File Reference

```
#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_date_time_types.hpp>
#include <stdair/basic/StructAbstract.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
```

Classes

- struct [TRADEMGEN::DemandCharacteristics](#)
Class modeling the characteristics of a demand type.

Namespaces

- namespace [TRADEMGEN](#)

25.40 DemandCharacteristics.hpp

```
00001 #ifndef __TRADEMGEN_BAS_DEMAND_CHARACTERISTICS_HPP
00002 #define __TRADEMGEN_BAS_DEMAND_CHARACTERISTICS_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_basic_types.hpp>
00011 #include <stdair/stdair_date_time_types.hpp>
00012 #include <stdair/basic/StructAbstract.hpp>
00013 // TraDemGen
00014 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00015
00016 namespace TRADEMGEN {
00017
00021 struct DemandCharacteristics : public stdair::StructAbstract {
00022
00023 public:
00024     // ////////////////////////////////// Business support methods //////////////////////////////////
00028     const stdair::AirportCode_T&
00029     getPOSValue (const stdair::Probability_T& iCumulativeProbability) const;
00030
00034     bool checkPOSValue (const stdair::AirportCode_T& iPOS) const;
00035
00036
00037 public:
00038     // ////////////////////////////////// Display support methods //////////////////////////////////
00042     const std::string describe() const;
00043
00044
00045 public:
00046     // ////////////////////////////////// Constructors and destructors //////////////////////////////////
00050     DemandCharacteristics (const ArrivalPatternCumulativeDistribution_T&,
00051                           const POSProbabilityMassFunction_T&,
00052                           const ChannelProbabilityMassFunction_T&,
00053                           const TripTypeProbabilityMassFunction_T&,
00054                           const StayDurationProbabilityMassFunction_T&,
00055                           const FrequentFlyerProbabilityMassFunction_T&,
00056                           const PreferredDepartureTimeContinuousDistribution_T
00057     &,
00058                           const stdair::WTP_T&,
00059                           const ValueOfTimeContinuousDistribution_T&);
00063     DemandCharacteristics ();
00064
00068     DemandCharacteristics (const DemandCharacteristics&);
00069
00073     ~DemandCharacteristics ();
00074
00075
00076 public:
00077     // ////////////////////////////////// Attributes //////////////////////////////////
```

```

00083     ContinuousFloatDuration_T _arrivalPattern;
00084
00088     POSProbabilityMass_T _posProbabilityMass;
00089
00093     ChannelProbabilityMass_T _channelProbabilityMass;
00094
00098     TripTypeProbabilityMass_T _tripTypeProbabilityMass;
00099
00103     StayDurationProbabilityMass_T _stayDurationProbabilityMass;
00104
00108     FrequentFlyerProbabilityMass_T _frequentFlyerProbabilityMass;
00109
00113     PreferredDepartureTimeCumulativeDistribution_T
    _preferredDepartureTimeCumulativeDistribution;
00114
00119     stdair::WTP_T _minWTP;
00120
00124     CumulativeDistribution_T _frat5Pattern;
00125
00129     ValueOfTimeCumulativeDistribution_T _valueOfTimeCumulativeDistribution;
00130 };
00131
00132 }
00133 #endif // __TRADEMGEN_BAS_DEMAND_CHARACTERISTICS_HPP

```

25.41 trademgen/basic/DemandCharacteristicsTypes.hpp File Reference

```

#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_date_time_types.hpp>
#include <stdair/stdair_demand_types.hpp>
#include <trademgen/basic/ContinuousAttributeLite.hpp>
#include <trademgen/basic/CategoricalAttributeLite.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef
ContinuousAttributeLite
< stdair::FloatDuration_T > [TRADEMGEN::ContinuousFloatDuration_T](#)
- typedef
ContinuousFloatDuration_T::ContinuousDistribution_T [TRADEMGEN::ArrivalPatternCumulativeDistribution_T](#)
- typedef
CategoricalAttributeLite
< stdair::AirportCode_T > [TRADEMGEN::POSProbabilityMass_T](#)
- typedef
POSProbabilityMass_T::ProbabilityMassFunction_T [TRADEMGEN::POSProbabilityMassFunction_T](#)
- typedef
CategoricalAttributeLite
< stdair::ChannelLabel_T > [TRADEMGEN::ChannelProbabilityMass_T](#)
- typedef
ChannelProbabilityMass_T::ProbabilityMassFunction_T [TRADEMGEN::ChannelProbabilityMassFunction_T](#)
- typedef
CategoricalAttributeLite
< stdair::TripType_T > [TRADEMGEN::TripTypeProbabilityMass_T](#)
- typedef
TripTypeProbabilityMass_T::ProbabilityMassFunction_T [TRADEMGEN::TripTypeProbabilityMassFunction_T](#)
- typedef
CategoricalAttributeLite
< stdair::DayDuration_T > [TRADEMGEN::StayDurationProbabilityMass_T](#)

- typedef
StayDurationProbabilityMass_T::ProbabilityMassFunction_T TRADEMGEN::StayDurationProbabilityMass-
Function_T
- typedef
CategoricalAttributeLite
< stdair::FrequentFlyer_T > TRADEMGEN::FrequentFlyerProbabilityMass_T
- typedef
FrequentFlyerProbabilityMass_T::ProbabilityMassFunction_T TRADEMGEN::FrequentFlyerProbabilityMass-
Function_T
- typedef
ContinuousAttributeLite
< stdair::IntDuration_T > TRADEMGEN::PreferredDepartureTimeCumulativeDistribution_T
- typedef
PreferredDepartureTimeCumulativeDistribution_T::ContinuousDistribution_T TRADEMGEN::Preferred-
DepartureTimeContinuousDistribution_T
- typedef
ContinuousAttributeLite
< stdair::PriceValue_T > TRADEMGEN::ValueOfTimeCumulativeDistribution_T
- typedef
ValueOfTimeCumulativeDistribution_T::ContinuousDistribution_T TRADEMGEN::ValueOfTimeContinuous-
Distribution_T
- typedef
ContinuousAttributeLite
< stdair::RealNumber_T > TRADEMGEN::CumulativeDistribution_T
- typedef
CumulativeDistribution_T::ContinuousDistribution_T TRADEMGEN::FRAT5Pattern_T

25.42 DemandCharacteristicsTypes.hpp

```

00001 #ifndef __TRADEMGEN_BAS_DEMANDCHARACTERISTICSTYPES_HPP
00002 #define __TRADEMGEN_BAS_DEMANDCHARACTERISTICSTYPES_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/stdair_date_time_types.hpp>
00010 #include <stdair/stdair_demand_types.hpp>
00011 // TraDemGen
00012 #include <trademgen/basic/ContinuousAttributeLite.hpp>
00013 #include <trademgen/basic/CategoricalAttributeLite.hpp>
00014
00015 namespace TRADEMGEN {
00016
00019     typedef ContinuousAttributeLite<stdair::FloatDuration_T>
ContinuousFloatDuration_T;
00020
00022     typedef ContinuousFloatDuration_T::ContinuousDistribution_T
ArrivalPatternCumulativeDistribution_T;
00023
00025     typedef CategoricalAttributeLite<stdair::AirportCode_T> POSProbabilityMass_T;
00026
00028     typedef POSProbabilityMass_T::ProbabilityMassFunction_T
POSProbabilityMassFunction_T;
00029
00031     typedef CategoricalAttributeLite<stdair::ChannelLabel_T>
ChannelProbabilityMass_T;
00032
00034     typedef ChannelProbabilityMass_T::ProbabilityMassFunction_T
ChannelProbabilityMassFunction_T;
00035
00037     typedef CategoricalAttributeLite<stdair::TripType_T> TripTypeProbabilityMass_T
;
00038
00040     typedef TripTypeProbabilityMass_T::ProbabilityMassFunction_T
TripTypeProbabilityMassFunction_T;
00041
00043     typedef CategoricalAttributeLite<stdair::DayDuration_T>
StayDurationProbabilityMass_T;
00044

```

```

00046     typedef StayDurationProbabilityMass_T::ProbabilityMassFunction_T
StayDurationProbabilityMassFunction_T;
00047
00049     typedef CategoricalAttributeLite<stdair::FrequentFlyer_T>
FrequentFlyerProbabilityMass_T;
00050
00052     typedef FrequentFlyerProbabilityMass_T::ProbabilityMassFunction_T
FrequentFlyerProbabilityMassFunction_T;
00053
00055     typedef ContinuousAttributeLite<stdair::IntDuration_T>
PreferredDepartureTimeCumulativeDistribution_T;
00056
00058     typedef
PreferredDepartureTimeCumulativeDistribution_T::ContinuousDistribution_T
PreferredDepartureTimeContinuousDistribution_T;
00059
00061     typedef ContinuousAttributeLite<stdair::PriceValue_T>
ValueOfTimeCumulativeDistribution_T;
00062
00064     typedef ValueOfTimeCumulativeDistribution_T::ContinuousDistribution_T
ValueOfTimeContinuousDistribution_T;
00065
00067     typedef ContinuousAttributeLite<stdair::RealNumber_T> CumulativeDistribution_T
;
00068     typedef CumulativeDistribution_T::ContinuousDistribution_T FRAT5Pattern_T;
00069 }
00070 #endif // __TRADEMGEN_BAS_DEMANDCHARACTERISTICSTYPES_HPP

```

25.43 trademgen/basic/DemandDistribution.cpp File Reference

```

#include <cassert>
#include <sstream>
#include <stdair/stdair_date_time_types.hpp>
#include <trademgen/basic/DemandDistribution.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.44 DemandDistribution.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // StdAir
00008 #include <stdair/stdair_date_time_types.hpp>
00009 // TraDemGen
00010 #include <trademgen/basic/DemandDistribution.hpp>
00011
00012 namespace TRADEMGEN {
00013
00014 // //////////////////////////////////////
00015 DemandDistribution::DemandDistribution (const stdair::NbOfRequests_T& iMean,
00016                                         const stdair::StdDevValue_T& iStdDev)
00017     : _meanNumberOfRequests (iMean),
00018       _stdDevNumberOfRequests (iStdDev) {
00019 }
00020
00021 // //////////////////////////////////////
00022 DemandDistribution::DemandDistribution() {
00023 }
00024
00025 // //////////////////////////////////////
00026 DemandDistribution::~DemandDistribution() {
00027 }
00028
00029 // //////////////////////////////////////
00030 DemandDistribution::
00031 DemandDistribution (const DemandDistribution& iDemandDistribution)
00032     : _meanNumberOfRequests (iDemandDistribution._meanNumberOfRequests),
00033       _stdDevNumberOfRequests (iDemandDistribution._stdDevNumberOfRequests) {
00034 }

```



```

00035
00036 // //////////////////////////////////////
00037 void DemandDistribution::fromStream (std::istream& ioIn) {
00038 }
00039
00040 // //////////////////////////////////////
00041 const std::string DemandDistribution::describe() const {
00042     std::ostringstream oStr;
00043     oStr << "N (" << _meanNumberOfRequests << ", "
00044         << _stdDevNumberOfRequests << ")";
00045     return oStr.str();
00046 }
00047
00048 // //////////////////////////////////////
00049 std::string DemandDistribution::display() const {
00050     std::ostringstream oStr;
00051     oStr << describe() << std::endl;
00052     return oStr.str();
00053 }
00054
00055 }
00056

```

25.45 trademgen/basic/DemandDistribution.hpp File Reference

```

#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/StructAbstract.hpp>
#include <trademgen/basic/ContinuousAttribute.hpp>

```

Classes

- struct [TRADEMGEN::DemandDistribution](#)
Class modeling the distribution of a demand type.

Namespaces

- namespace [TRADEMGEN](#)

25.46 DemandDistribution.hpp

```

00001 #ifndef __TRADEMGEN_BAS_DEMAND_DISTRIBUTION_HPP
00002 #define __TRADEMGEN_BAS_DEMAND_DISTRIBUTION_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_basic_types.hpp>
00011 #include <stdair/basic/StructAbstract.hpp>
00012 // TraDemGen
00013 #include <trademgen/basic/ContinuousAttribute.hpp>
00014
00015 namespace TRADEMGEN {
00016
00020 struct DemandDistribution : public stdair::StructAbstract {
00021 public:
00022     // ////////////////////////////////// Constructors and destructors //////////////////////////////////
00026     DemandDistribution (const stdair::NbOfRequests_T& iMean,
00027         const stdair::StdDevValue_T& iStdDev);
00031     DemandDistribution();
00035     DemandDistribution (const DemandDistribution&);
00039     ~DemandDistribution();
00040
00041
00042 public:
00043     // ////////////////////////////////// Display Support Methods //////////////////////////////////
00049     void fromStream (std::istream& ioIn);
00050

```

```

00054     const std::string describe() const;
00055
00059     std::string display() const;
00060
00061
00062 public:
00063     // ////////// Attributes //////////
00067     stdair::NbOfRequests_T _meanNumberOfRequests;
00068
00072     stdair::StdDevValue_T _stdDevNumberOfRequests;
00073 };
00074
00075 }
00076 #endif // __TRADEMGEN_BAS_DEMAND_DISTRIBUTION_HPP

```

25.47 trademgen/basic/DictionaryManager.cpp File Reference

```
#include <trademgen/basic/DictionaryManager.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)

25.48 DictionaryManager.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // TRADEMGEN
00005 #include <trademgen/basic/DictionaryManager.hpp>
00006
00007 namespace TRADEMGEN {
00008     // //////////////////////////////////////
00009     const stdair::Probability_T DictionaryManager::
00010     keyToValue (const DictionaryKey_T iKey) {
00011         // return static_cast<stdair::Probability_T>(iKey) / 1000;
00012         return iKey;
00013     }
00014
00015     // //////////////////////////////////////
00016     const DictionaryKey_T DictionaryManager::
00017     valueToKey (const stdair::Probability_T iValue) {
00018         // return iValue * 1000;
00019         return iValue;
00020     }
00021 }

```

25.49 trademgen/basic/DictionaryManager.hpp File Reference

```
#include <stdair/stdair_maths_types.hpp>
```

Classes

- class [TRADEMGEN::DictionaryManager](#)
Class wrapper of dictionary business methods.

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef stdair::Probability_T [TRADEMGEN::DictionaryKey_T](#)

25.50 DictionaryManager.hpp

```

00001 ///////////////////////////////////////////////////////////////////
00002 #ifndef __TRADEMGEN_BASIC_DICTIONARYMANAGER_HPP
00003 #define __TRADEMGEN_BASIC_DICTIONARYMANAGER_HPP
00004
00005 ///////////////////////////////////////////////////////////////////
00006 // Import section
00007 ///////////////////////////////////////////////////////////////////
00008 // StdAir
00009 #include <stdair/stdair_maths_types.hpp>
00010
00011 namespace TRADEMGEN {
00012
00013     /////////////////////////////////////////////////////////////////// Type definitions ///////////////////////////////////////////////////////////////////
00014     //typedef unsigned short DictionaryKey_T;
00015     typedef stdair::Probability_T DictionaryKey_T;
00016
00017     class DictionaryManager {
00018     public:
00019         /////////////////////////////////////////////////////////////////// Business methods ///////////////////////////////////////////////////////////////////
00020         static const stdair::Probability_T keyToValue (const DictionaryKey_T);
00021
00022         static const DictionaryKey_T valueToKey (const stdair::Probability_T);
00023     };
00024 }
00025 #endif // __TRADEMGEN_BASIC_DICTIONARYMANAGER_HPP

```

25.51 trademgen/basic/RandomGenerationContext.cpp File Reference

```

#include <cassert>
#include <sstream>
#include <trademgen/basic/RandomGenerationContext.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.52 RandomGenerationContext.cpp

```

00001 ///////////////////////////////////////////////////////////////////
00002 // Import section
00003 ///////////////////////////////////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // TraDemGen
00008 #include <trademgen/basic/RandomGenerationContext.hpp>
00009
00010 namespace TRADEMGEN {
00011
00012     ///////////////////////////////////////////////////////////////////
00013     RandomGenerationContext::RandomGenerationContext ()
00014         : _numberOfRequestsGeneratedSoFar (0),
00015         _cumulativeProbabilitySoFar (0.0) {
00016     }
00017
00018     ///////////////////////////////////////////////////////////////////
00019     RandomGenerationContext::
00020     RandomGenerationContext (const RandomGenerationContext& iRGC)
00021         : _numberOfRequestsGeneratedSoFar (iRGC._numberOfRequestsGeneratedSoFar),
00022         _cumulativeProbabilitySoFar (iRGC._cumulativeProbabilitySoFar) {
00023     }
00024
00025     ///////////////////////////////////////////////////////////////////
00026     RandomGenerationContext::~RandomGenerationContext () {
00027     }
00028
00029     ///////////////////////////////////////////////////////////////////
00030     const std::string RandomGenerationContext::describe() const {
00031         std::ostringstream oStr;
00032         oStr << _numberOfRequestsGeneratedSoFar
00033             << " => " << _cumulativeProbabilitySoFar;
00034         return oStr.str();
00035     }

```

```

00036
00037 //////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
00038 void RandomGenerationContext::incrementGeneratedRequestsCounter() {
00039     ++_numberOfRequestsGeneratedSoFar;
00040 }
00041
00042 //////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
00043 void RandomGenerationContext::reset() {
00044     _cumulativeProbabilitySoFar = 0.0;
00045     _numberOfRequestsGeneratedSoFar = 0;
00046 }
00047
00048 }

```

25.53 trademgen/basic/RandomGenerationContext.hpp File Reference

```

#include <iosfwd>
#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_maths_types.hpp>
#include <stdair/basic/StructAbstract.hpp>

```

Classes

- struct [TRADEMGEN::RandomGenerationContext](#)

Namespaces

- namespace [TRADEMGEN](#)

25.54 RandomGenerationContext.hpp

```

00001 #ifndef __TRADEMGEN_BAS_RANDOM_GENERATION_CONTEXT_HPP
00002 #define __TRADEMGEN_BAS_RANDOM_GENERATION_CONTEXT_HPP
00003
00004 //////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
00005 // Import section
00006 //////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
00007 // STL
00008 #include <iosfwd>
00009 #include <string>
00010 // StdAir
00011 #include <stdair/stdair_basic_types.hpp>
00012 #include <stdair/stdair_maths_types.hpp>
00013 #include <stdair/basic/StructAbstract.hpp>
00014
00015 namespace TRADEMGEN {
00016
00020 struct RandomGenerationContext : public stdair::StructAbstract {
00021 public:
00022     ////////////////////////////////////////////////////////////////// Getters ///////////////////////////////////////////////////
00026     const stdair::Count_T& getNumberOfRequestsGeneratedSoFar() const {
00027         return _numberOfRequestsGeneratedSoFar;
00028     }
00029
00034     const stdair::Probability_T& getCumulativeProbabilitySoFar() const {
00035         return _cumulativeProbabilitySoFar;
00036     }
00037
00038 public:
00039     ////////////////////////////////////////////////////////////////// Setters ///////////////////////////////////////////////////
00043     void setNumberOfRequestsGeneratedSoFar (const stdair::Count_T& iCount) {
00044         _numberOfRequestsGeneratedSoFar = iCount;
00045     }
00046
00051     void setCumulativeProbabilitySoFar (const stdair::Probability_T& iProb) {
00052         _cumulativeProbabilitySoFar = iProb;
00053     }
00054
00055 public:
00056

```

```

00057 // ////////// Constructors and destructors //////////
00061 RandomGenerationContext();
00062
00066 RandomGenerationContext (const RandomGenerationContext&);
00067
00071 ~RandomGenerationContext();
00072
00073
00074 public:
00075 // ////////// Business Methods //////////
00079 void incrementGeneratedRequestsCounter();
00080
00084 void reset();
00085
00086
00087 public:
00088 // ////////// Display Support Methods //////////
00092 const std::string describe() const;
00093
00094
00095 private:
00096 // ////////// Attributes //////////
00100 stdair::Count_T _numberOfRequestsGeneratedSoFar;
00101
00106 stdair::Probability_T _cumulativeProbabilitySoFar;
00107 };
00108
00109 }
00110 #endif // __STDAIR_BAS_RANDOM_GENERATION_CONTEXT_HPP

```

25.55 trademgen/batches/trademgen.cpp File Reference

```

#include <cassert>
#include <sstream>
#include <fstream>
#include <vector>
#include <list>
#include <string>
#include <boost/tokenizer.hpp>
#include <boost/program_options.hpp>
#include <boost/accumulators/accumulators.hpp>
#include <boost/accumulators/statistics.hpp>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/basic/ProgressStatusSet.hpp>
#include <stdair/basic/DemandGenerationMethod.hpp>
#include <stdair/bom/EventStruct.hpp>
#include <stdair/bom/EventQueue.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/bom/BomDisplay.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/TRADEMGEN_Service.hpp>
#include <trademgen/config/trademgen-paths.hpp>

```

Typedefs

- typedef unsigned int [NbOfRuns_T](#)
- typedef `ba::accumulator_set`
`< double, ba::stats`
`< ba::tag::min, ba::tag::max,`
`ba::tag::mean(ba::immediate),`
`ba::tag::sum,`
`ba::tag::variance > > stat_acc_type`

Functions

- const stdair::Filename_T [K_TRADEMGEN_DEFAULT_LOG_FILENAME](#) ("trademgen.log")
- const stdair::Filename_T [K_TRADEMGEN_DEFAULT_INPUT_FILENAME](#) (STDAIR_SAMPLE_DIR"/demand01.csv")
- const stdair::Filename_T [K_TRADEMGEN_DEFAULT_OUTPUT_FILENAME](#) ("request.csv")
- void [stat_display](#) (std::ostream &ostream, const [stat_acc_type](#) &iStatAcc)
- template<class T >
std::ostream & [operator<<](#) (std::ostream &os, const std::vector< T > &v)
- int [readConfiguration](#) (int argc, char *argv[], bool &iolsBuiltin, stdair::RandomSeed_T &ioRandomSeed, [NbOfRuns_T](#) &ioRandomRuns, stdair::Filename_T &ioInputFilename, stdair::Filename_T &ioOutputFilename, stdair::Filename_T &ioLogFilename, stdair::DemandGenerationMethod &ioDemandGenerationMethod)
- void [generateDemand](#) ([TRADEMGEN::TRADEMGEN_Service](#) &ioTrademgenService, const stdair::Filename_T &ioOutputFilename, const [NbOfRuns_T](#) &iNbOfRuns, const stdair::DemandGenerationMethod &iDemandGenerationMethod)
- int [main](#) (int argc, char *argv[])

Variables

- const
stdair::DemandGenerationMethod [K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD](#)
- const char [K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR](#)
- const stdair::RandomSeed_T [K_TRADEMGEN_DEFAULT_RANDOM_SEED](#)
- const [NbOfRuns_T](#) [K_TRADEMGEN_DEFAULT_RANDOM_DRAWNS](#) = 1
- const bool [K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT](#) = false
- const int [K_TRADEMGEN_EARLY_RETURN_STATUS](#) = 99

25.55.1 Typedef Documentation

25.55.1.1 typedef unsigned int [NbOfRuns_T](#)

Definition at line 40 of file [trademgen.cpp](#).

25.55.1.2 typedef ba::accumulator_set<double, ba::stats<ba::tag::min, ba::tag::max, ba::tag::mean (ba::immediate), ba::tag::sum, ba::tag::variance> > [stat_acc_type](#)

Type definition to gather statistics.

Definition at line 49 of file [trademgen.cpp](#).

25.55.2 Function Documentation

25.55.2.1 const stdair::Filename_T [K_TRADEMGEN_DEFAULT_LOG_FILENAME](#) ("trademgen.log")

Default name and location for the log file.

Referenced by [readConfiguration\(\)](#).

25.55.2.2 const stdair::Filename_T [K_TRADEMGEN_DEFAULT_INPUT_FILENAME](#) (STDAIR_SAMPLE_DIR"/demand01.csv")

Default name and location for the (CSV) input file.

Referenced by [readConfiguration\(\)](#).

25.55.2.3 const stdair::Filename_T [K_TRADEMGEN_DEFAULT_OUTPUT_FILENAME](#) ("request.csv")

Default name and location for the (CSV) output file.

Referenced by [readConfiguration\(\)](#).

25.55.2.4 `void stat_display (std::ostream & oStream, const stat_acc_type & iStatAcc)`

Display the statistics held by the dedicated accumulator.

Definition at line 107 of file [trademgen.cpp](#).

Referenced by [generateDemand\(\)](#).

25.55.2.5 `template<class T> std::ostream& operator<< (std::ostream & os, const std::vector< T> & v)`

Definition at line 129 of file [trademgen.cpp](#).

25.55.2.6 `int readConfiguration (int argc, char * argv[], bool & iolsBuiltin, stdair::RandomSeed.T & ioRandomSeed, NbOfRuns_T & ioRandomRuns, stdair::Filename_T & ioInputFilename, stdair::Filename_T & ioOutputFilename, stdair::Filename_T & ioLogFilename, stdair::DemandGenerationMethod & ioDemandGenerationMethod)`

Read and parse the command line options.

Definition at line 138 of file [trademgen.cpp](#).

References [K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT](#), [K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR](#), [K_TRADEMGEN_DEFAULT_INPUT_FILENAME\(\)](#), [K_TRADEMGEN_DEFAULT_LOG_FILENAME\(\)](#), [K_TRADEMGEN_DEFAULT_OUTPUT_FILENAME\(\)](#), [K_TRADEMGEN_EARLY_RETURN_STATUS](#), [PACKAGE_NAME](#), [PACKAGE_VERSION](#), and [PREFIXDIR](#).

Referenced by [main\(\)](#).

25.55.2.7 `void generateDemand (TRADEMGEN::TRADEMGEN_Service & ioTrademgenService, const stdair::Filename_T & iOutputFilename, const NbOfRuns_T & iNbOfRuns, const stdair::DemandGenerationMethod & iDemandGenerationMethod)`

Definition at line 278 of file [trademgen.cpp](#).

References [TRADEMGEN::TRADEMGEN_Service::csvDisplay\(\)](#), [TRADEMGEN::TRADEMGEN_Service::generateFirstRequests\(\)](#), [TRADEMGEN::TRADEMGEN_Service::generateNextRequest\(\)](#), [TRADEMGEN::TRADEMGEN_Service::getExpectedTotalNumberOfRequestsToBeGenerated\(\)](#), [TRADEMGEN::TRADEMGEN_Service::isQueueDone\(\)](#), [TRADEMGEN::TRADEMGEN_Service::popEvent\(\)](#), [TRADEMGEN::TRADEMGEN_Service::reset\(\)](#), and [stat_display\(\)](#).

Referenced by [main\(\)](#), and [TRADEMGEN::TRADEMGEN_Service::parseAndLoad\(\)](#).

25.55.2.8 `int main (int argc, char * argv[])`

Definition at line 422 of file [trademgen.cpp](#).

References [TRADEMGEN::TRADEMGEN_Service::buildSampleBom\(\)](#), [generateDemand\(\)](#), [K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD](#), [TRADEMGEN::TRADEMGEN_Service::parseAndLoad\(\)](#), and [readConfiguration\(\)](#).

25.55.3 Variable Documentation

25.55.3.1 `const stdair::DemandGenerationMethod K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD`

Initial value:

```
stdair::DemandGenerationMethod::POI_PRO
```

Default demand generation method: Poisson Process.

Definition at line 72 of file [trademgen.cpp](#).

Referenced by [main\(\)](#).

25.55.3.2 const char K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR

Initial value:

```
K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD.getMethodAsChar()
```

Default demand generation method name: 'P' for Poisson Process.

Definition at line 78 of file [trademngen.cpp](#).

Referenced by [readConfiguration\(\)](#).

25.55.3.3 const stdair::RandomSeed_T K_TRADEMGEN_DEFAULT_RANDOM_SEED

Initial value:

```
stdair::DEFAULT_RANDOM_SEED
```

Default random generation seed (e.g., 120765987).

Definition at line 84 of file [trademngen.cpp](#).

Referenced by [readConfiguration\(\)](#).

25.55.3.4 const NbOfRuns_T K_TRADEMGEN_DEFAULT_RANDOM_DRAWS = 1

Default number of random draws to be generated (best if over 100).

Definition at line 90 of file [trademngen.cpp](#).

25.55.3.5 const bool K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT = false

Default for the input type. It can be either built-in or provided by an input file. That latter must then be given with the -i option.

Definition at line 96 of file [trademngen.cpp](#).

Referenced by [readConfiguration\(\)](#).

25.55.3.6 const int K_TRADEMGEN_EARLY_RETURN_STATUS = 99

Early return status (so that it can be differentiated from an error).

Definition at line 101 of file [trademngen.cpp](#).

Referenced by [main\(\)](#), and [readConfiguration\(\)](#).

25.56 trademngen.cpp

```
00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 #include <fstream>
00008 #include <vector>
00009 #include <list>
00010 #include <string>
00011 // /// Boost (Extended STL) ///
00012 // Boost Tokeniser
00013 #include <boost/tokenizer.hpp>
00014 // Boost Program Options
00015 #include <boost/program_options.hpp>
00016 // Boost Accumulators
00017 #include <boost/accumulators/accumulators.hpp>
00018 #include <boost/accumulators/statistics.hpp>
00019 // Boost Progress
```



```

00020 // #include <boost/progress.hpp>
00021 // StdAir
00022 #include <stdair/stdair_basic_types.hpp>
00023 #include <stdair/basic/BasConst_General.hpp>
00024 #include <stdair/basic/ProgressStatusSet.hpp>
00025 #include <stdair/basic/DemandGenerationMethod.hpp>
00026 #include <stdair/bom/EventStruct.hpp>
00027 #include <stdair/bom/EventQueue.hpp>
00028 #include <stdair/bom/BookingRequestStruct.hpp>
00029 #include <stdair/bom/BomDisplay.hpp>
00030 #include <stdair/service/Logger.hpp>
00031 // TraDemGen
00032 #include <trademgen/TRADEMGEN_Service.hpp>
00033 #include <trademgen/config/trademgen-paths.hpp>
00034 #include <trademgen/config/trademgen-paths.hpp>
00035
00036 // Aliases for namespaces
00037 namespace ba = boost::accumulators;
00038
00039 // ////////// Specific type definitions //////////
00040 typedef unsigned int NbOfRuns_T;
00041
00042 typedef ba::accumulator_set<double,
00043                             ba::stats<ba::tag::min, ba::tag::max,
00044                                     ba::tag::mean (ba::immediate),
00045                                     ba::tag::sum,
00046                                     ba::tag::variance> > stat_acc_type;
00047
00048 // ////////// Constants //////////
00049 const stdair::Filename_T K_TRADEMGEN_DEFAULT_LOG_FILENAME ("trademgen.log");
00050
00051 const stdair::Filename_T K_TRADEMGEN_DEFAULT_INPUT_FILENAME (STDAIR_SAMPLE_DIR
00052                                                                "/demand01.csv");
00053
00054 const stdair::Filename_T K_TRADEMGEN_DEFAULT_OUTPUT_FILENAME ("request.csv");
00055
00056 const stdair::DemandGenerationMethod
00057 K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD =
00058     stdair::DemandGenerationMethod::POI_PRO;
00059
00060 const char K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR =
00061     K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD.getMethodAsChar();
00062
00063 const stdair::RandomSeed_T K_TRADEMGEN_DEFAULT_RANDOM_SEED =
00064     stdair::DEFAULT_RANDOM_SEED;
00065
00066 const NbOfRuns_T K_TRADEMGEN_DEFAULT_RANDOM_DRAWS = 1;
00067
00068 const bool K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT = false;
00069
00070 const int K_TRADEMGEN_EARLY_RETURN_STATUS = 99;
00071
00072 void stat_display (std::ostream& oStream, const stat_acc_type& iStatAcc) {
00073     // Store current formatting flags of the output stream
00074     std::ios::fmtflags oldFlags = oStream.flags();
00075
00076     //
00077     oStream.setf (std::ios::fixed);
00078
00079     //
00080     oStream << "Statistics for the demand generation runs: " << std::endl;
00081     oStream << "  minimum   = " << ba::min (iStatAcc) << std::endl;
00082     oStream << "    mean    = " << ba::mean (iStatAcc) << std::endl;
00083     oStream << " maximum   = " << ba::max (iStatAcc) << std::endl;
00084     oStream << "   count    = " << ba::count (iStatAcc) << std::endl;
00085     oStream << " variance  = " << ba::variance (iStatAcc) << std::endl;
00086
00087     // Reset formatting flags of output stream
00088     oStream.flags (oldFlags);
00089 }
00090
00091 // ////////// Parsing of Options & Configuration //////////
00092 // A helper function to simplify the main part.
00093 template<class T> std::ostream& operator<< (std::ostream& os,
00094                                             const std::vector<T>& v) {
00095     std::copy (v.begin(), v.end(), std::ostream_iterator<T> (std::cout, " "));
00096     return os;
00097 }
00098
00099 int readConfiguration (int argc, char* argv[], bool& ioIsBuiltin,
00100                       stdair::RandomSeed_T& ioRandomSeed,
00101                       NbOfRuns_T& ioRandomRuns,
00102                       stdair::Filename_T& ioInputFilename,
00103                       stdair::Filename_T& ioOutputFilename,
00104                       stdair::Filename_T& ioLogFilename,

```

```

00144         stdair::DemandGenerationMethod& ioDemandGenerationMethod
00145     ) {
00146         // Demand generation method as a single char (e.g., 'P' or 'S').
00147         char lDemandGenerationMethodChar;
00148
00149         // Default for the built-in input
00150         ioIsBuiltin = K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT;
00151
00152         // Declare a group of options that will be allowed only on command line
00153         boost::program_options::options_description generic ("Generic options");
00154         generic.add_options()
00155             ("prefix", "print installation prefix")
00156             ("version,v", "print version string")
00157             ("help,h", "produce help message");
00158
00159         // Declare a group of options that will be allowed both on command
00160         // line and in config file
00161         boost::program_options::options_description config ("Configuration");
00162         config.add_options()
00163             ("builtin,b",
00164              "The sample BOM tree can be either built-in or parsed from an input file.
00165              That latter must then be given with the -i/--input option")
00166             ("seed,s",
00167              boost::program_options::value<stdair::RandomSeed_T>(&ioRandomSeed)->
00168              default_value(K_TRADEMGEN_DEFAULT_RANDOM_SEED),
00169              "Seed for the random generation")
00170             ("draws,d",
00171              boost::program_options::value<NbOfRuns_T>(&ioRandomRuns)->default_value(
00172              K_TRADEMGEN_DEFAULT_RANDOM_DRAWS),
00173              "Number of runs for the demand generations")
00174             ("demandgeneration,G",
00175              boost::program_options::value< char >(&lDemandGenerationMethodChar)->
00176              default_value(K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD_CHAR),
00177              "Method used to generate the demand (i.e., the booking requests): Poisson
00178              Process (P) or Order Statistics (S)")
00179             ("input,i",
00180              boost::program_options::value< std::string >(&ioInputFilename)->
00181              default_value(K_TRADEMGEN_DEFAULT_INPUT_FILENAME),
00182              "(CSV) input file for the demand distributions")
00183             ("output,o",
00184              boost::program_options::value< std::string >(&ioOutputFilename)->
00185              default_value(K_TRADEMGEN_DEFAULT_OUTPUT_FILENAME),
00186              "(CSV) output file for the generated requests")
00187             ("log,l",
00188              boost::program_options::value< std::string >(&ioLogFilename)->
00189              default_value(K_TRADEMGEN_DEFAULT_LOG_FILENAME),
00190              "Filepath for the logs")
00191             ;
00192
00193         // Hidden options, will be allowed both on command line and
00194         // in config file, but will not be shown to the user.
00195         boost::program_options::options_description hidden ("Hidden options");
00196         hidden.add_options()
00197             ("copyright",
00198              boost::program_options::value< std::vector<std::string> >(),
00199              "Show the copyright (license)");
00200
00201         boost::program_options::options_description cmdline_options;
00202         cmdline_options.add(generic).add(config).add(hidden);
00203
00204         boost::program_options::options_description config_file_options;
00205         config_file_options.add(config).add(hidden);
00206
00207         boost::program_options::options_description visible ("Allowed options");
00208         visible.add(generic).add(config);
00209
00210         boost::program_options::positional_options_description p;
00211         p.add ("copyright", -1);
00212
00213         boost::program_options::variables_map vm;
00214         boost::program_options::
00215             store (boost::program_options::command_line_parser (argc, argv).
00216                   options (cmdline_options).positional(p).run(), vm);
00217
00218         std::ifstream ifs ("trademgen.cfg");
00219         boost::program_options::store (parse_config_file (ifs, config_file_options),
00220                                       vm);
00221         boost::program_options::notify (vm);
00222
00223         if (vm.count ("help")) {
00224             std::cout << visible << std::endl;
00225             return K_TRADEMGEN_EARLY_RETURN_STATUS;
00226         }
00227
00228         if (vm.count ("version")) {
00229             std::cout << PACKAGE_NAME << " , version " << PACKAGE_VERSION << std::endl;
00230         }
00231     }

```

```

00222     return K_TRADEMGEN_EARLY_RETURN_STATUS;
00223 }
00224
00225 if (vm.count ("prefix")) {
00226     std::cout << "Installation prefix: " << PREFIXDIR << std::endl;
00227     return K_TRADEMGEN_EARLY_RETURN_STATUS;
00228 }
00229
00230 if (vm.count ("builtin")) {
00231     ioIsBuiltin = true;
00232 }
00233 const std::string isBuiltinStr = (ioIsBuiltin == true)? "yes": "no";
00234 std::cout << "The BOM should be built-in? " << isBuiltinStr << std::endl;
00235
00236 if (ioIsBuiltin == false) {
00237     // The BOM tree should be built from parsing a demand input file
00238     if (vm.count ("input")) {
00239         ioInputFilename = vm["input"].as< std::string >();
00240         std::cout << "Input filename is: " << ioInputFilename << std::endl;
00241     } else {
00242         // The built-in option is not selected. However, no demand input file
00243         // is specified
00244         std::cerr << "Either one among the -b/--builtin and -i/--input "
00245             << "options must be specified" << std::endl;
00246     }
00247 }
00248
00249 if (vm.count ("output")) {
00250     ioOutputFilename = vm["output"].as< std::string >();
00251     std::cout << "Output filename is: " << ioOutputFilename << std::endl;
00252 }
00253
00254 if (vm.count ("log")) {
00255     ioLogFilename = vm["log"].as< std::string >();
00256     std::cout << "Log filename is: " << ioLogFilename << std::endl;
00257 }
00258
00259 if (vm.count ("demandgeneration")) {
00260     ioDemandGenerationMethod =
00261         stdair::DemandGenerationMethod (lDemandGenerationMethodChar);
00262     std::cout << "Date-time request generation method is: "
00263         << ioDemandGenerationMethod.describe() << std::endl;
00264 }
00265
00266 //
00267 std::cout << "The random generation seed is: " << ioRandomSeed << std::endl;
00268
00269 //
00270 std::cout << "The number of runs is: " << ioRandomRuns << std::endl;
00271
00272 return 0;
00273 }
00274
00275 ///////////////////////////////////////////////////////////////////
00276 void generateDemand (TRADEMGEN::TRADEMGEN_Service& ioTrademgenService,
00277     const stdair::Filename_T& iOutputFilename,
00278     const NbOfRuns_T& iNbOfRuns,
00279     const stdair::DemandGenerationMethod&
00280     iDemandGenerationMethod) {
00281
00282     // Open and clean the .csv output file
00283     std::ofstream output;
00284     output.open (iOutputFilename.c_str());
00285     output.clear();
00286
00287     // Initialise the statistics collector/accumulator
00288     stat_acc_type lStatAccumulator;
00289
00290     // Retrieve the expected (mean value of the) number of events to be
00291     // generated
00292     const stdair::Count_T& lExpectedNbOfEventsToBeGenerated =
00293         ioTrademgenService.getExpectedTotalNumberOfRequestsToBeGenerated();
00294
00295     // Initialise the (Boost) progress display object
00296     boost::progress_display lProgressDisplay (lExpectedNbOfEventsToBeGenerated
00297         * iNbOfRuns);
00298
00299     for (NbOfRuns_T runIdx = 1; runIdx <= iNbOfRuns; ++runIdx) {
00300         ///////////////////////////////////////////////////////////////////
00301         output << "Run number: " << runIdx << std::endl;
00302
00303         const stdair::Count_T& lActualNbOfEventsToBeGenerated =
00304             ioTrademgenService.generateFirstRequests (iDemandGenerationMethod);
00305
00306         // DEBUG
00307     }
00308 }

```

```

00312     STDAIR_LOG_DEBUG ("[" << runIdx << "] Expected: "
00313                     << lExpectedNbOfEventsToBeGenerated << ", actual: "
00314                     << lActualNbOfEventsToBeGenerated);
00315
00323     while (ioTrademgenService.isQueueDone() == false) {
00324
00325         // Extract the next event from the event queue
00326         stdair::EventStruct lEventStruct;
00327         stdair::ProgressStatusSet lProgressStatusSet =
00328             ioTrademgenService.popEvent (lEventStruct);
00329
00330         // DEBUG
00331         // STDAIR_LOG_DEBUG ("[" << runIdx << "] Poped event: '"
00332         //                     << lEventStruct.describe() << "'.");
00333
00334         // Extract the corresponding demand/booking request
00335         const stdair::BookingRequestStruct& lPoppedRequest =
00336             lEventStruct.getBookingRequest();
00337
00338         // DEBUG
00339         STDAIR_LOG_DEBUG ("[" << runIdx << "] Poped booking request: '"
00340                         << lPoppedRequest.describe() << "'.");
00341
00342         // Dump the request into the dedicated CSV file
00343         // stdair::BomDisplay::csvDisplay (output, lPoppedRequest);
00344
00345         // Retrieve the corresponding demand stream key
00346         const stdair::DemandGeneratorKey_T& lDemandStreamKey =
00347             lPoppedRequest.getDemandGeneratorKey();
00348
00349         // Assess whether more events should be generated for that demand stream
00350         const bool stillHavingRequestsToBeGenerated = ioTrademgenService.
00351             stillHavingRequestsToBeGenerated (lDemandStreamKey,
00352                                             lProgressStatusSet,
00353                                             iDemandGenerationMethod);
00354
00355         // DEBUG
00356         STDAIR_LOG_DEBUG (lProgressStatusSet.describe());
00357         STDAIR_LOG_DEBUG ("=> [" << lDemandStreamKey << "] is now processed. "
00358                         << "Still generate events for that demand stream? "
00359                         << stillHavingRequestsToBeGenerated);
00360
00361         // If there are still events to be generated for that demand stream,
00362         // generate and add them to the event queue
00363         if (stillHavingRequestsToBeGenerated == true) {
00364
00365             stdair::BookingRequestPtr_T lNextRequest_ptr =
00366                 ioTrademgenService.generateNextRequest (lDemandStreamKey,
00367                                                         iDemandGenerationMethod);
00368
00369             assert (lNextRequest_ptr != NULL);
00370
00371             // Sanity check
00372             const stdair::Duration_T lDuration =
00373                 lNextRequest_ptr->getRequestDateTime()
00374                 - lPoppedRequest.getRequestDateTime();
00375             if (lDuration.total_milliseconds() < 0) {
00376                 STDAIR_LOG_ERROR ("[" << lDemandStreamKey
00377                                 << "] The date-time of the generated event ("
00378                                 << lNextRequest_ptr->getRequestDateTime()
00379                                 << ") is lower than the date-time "
00380                                 << "of the current event ("
00381                                 << lPoppedRequest.getRequestDateTime() << ")");
00382                 assert (false);
00383             }
00384
00385             // DEBUG
00386             STDAIR_LOG_DEBUG ("[" << lDemandStreamKey << "] Added request: '"
00387                             << lNextRequest_ptr->describe()
00388                             << "'. Is queue done? "
00389                             << ioTrademgenService.isQueueDone());
00390         }
00391         // DEBUG
00392         STDAIR_LOG_DEBUG ("");
00393
00394         // Update the progress display
00395         ++lProgressDisplay;
00396     }
00397
00398     // Add the number of events to the statistics accumulator
00399     lStatAccumulator (lActualNbOfEventsToBeGenerated);
00400
00401     // Reset the service (including the event queue) for the next run
00402     ioTrademgenService.reset();
00403 }
00404
00405 // DEBUG

```

```

00406     STDAIR_LOG_DEBUG ("End of the demand generation. Following are some "
00407                       "statistics for the " << iNbOfRuns << " runs.");
00408     std::ostringstream oStatStr;
00409     stat_display (oStatStr, lStatAccumulator);
00410     STDAIR_LOG_DEBUG (oStatStr.str());
00411
00412     // DEBUG
00413     const std::string& lBOMStr = ioTrademgenService.csvDisplay();
00414     STDAIR_LOG_DEBUG (lBOMStr);
00415
00416     // Close the output file
00417     output.close();
00418 }
00419
00420
00421 // ////////////////////////////////// M A I N //////////////////////////////////
00422 int main (int argc, char* argv[]) {
00423
00424     // State whether the BOM tree should be built-in or parsed from an input file
00425     bool isBuiltin;
00426
00427     // Random generation seed
00428     stdair::RandomSeed_T lRandomSeed;
00429
00430     // Number of random draws to be generated (best if greater than 100)
00431     NbOfRuns_T lNbOfRuns;
00432
00433     // Input file name
00434     stdair::Filename_T lInputFilename;
00435
00436     // Output file name
00437     stdair::Filename_T lOutputFilename;
00438
00439     // Output log File
00440     stdair::Filename_T lLogFilename;
00441
00442     // Demand generation method.
00443     stdair::DemandGenerationMethod
00444         lDemandGenerationMethod (K_TRADEMGEN_DEFAULT_DEMAND_GENERATION_METHOD);
00445
00446     // Call the command-line option parser
00447     const int lOptionParserStatus =
00448         readConfiguration (argc, argv, isBuiltin, lRandomSeed, lNbOfRuns,
00449                             lInputFilename, lOutputFilename, lLogFilename,
00450                             lDemandGenerationMethod);
00451
00452     if (lOptionParserStatus == K_TRADEMGEN_EARLY_RETURN_STATUS) {
00453         return 0;
00454     }
00455
00456     // Set the log parameters
00457     std::ofstream logOutputFile;
00458     // Open and clean the log outputfile
00459     logOutputFile.open (lLogFilename.c_str());
00460     logOutputFile.clear();
00461
00462     // Set up the log parameters
00463     const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
00464
00465     // Initialise the TraDemGen service object
00466     TRADEMGEN::TRADEMGEN_Service trademgenService (lLogParams, lRandomSeed);
00467
00468     // Check whether or not a (CSV) input file should be read
00469     if (isBuiltin == true) {
00470         // Create a sample DemandStream object, and insert it within the BOM tree
00471         trademgenService.buildSampleBom();
00472     } else {
00473         // Create the DemandStream objects, and insert them within the BOM tree
00474         trademgenService.parseAndLoad (lInputFilename);
00475     }
00476
00477     // Calculate the expected number of events to be generated.
00478     generateDemand (trademgenService, lOutputFilename, lNbOfRuns,
00479                     lDemandGenerationMethod);
00480
00481     // Close the Log outputFile
00482     logOutputFile.close();
00483
00484     /*
00485     \note: as that program is not intended to be run on a server in
00486     production, it is better not to catch the exceptions. When it
00487     happens (that an exception is throwned), that way we get the
00488     call stack.
00489     */
00490
00491     return 0;
00492 }

```

```
00493 }
```

25.57 trademgen/ui/qt/trademgen/trademgen.cpp File Reference

```
#include "trademgen.h"
#include <QtGui/QLabel>
#include <QtGui/QMenu>
#include <QtGui/QMenuBar>
#include <QtGui/QAction>
#include "trademgen.moc"
```

25.58 trademgen.cpp

```
00001 #include "trademgen.h"
00002
00003 #include <QtGui/QLabel>
00004 #include <QtGui/QMenu>
00005 #include <QtGui/QMenuBar>
00006 #include <QtGui/QAction>
00007
00008 trademgen::trademgen()
00009 {
00010     QLabel* l = new QLabel( this );
00011     l->setText( "Hello World!" );
00012     setCentralWidget( l );
00013     QAction* a = new QAction(this);
00014     a->setText( "Quit" );
00015     connect(a, SIGNAL(triggered()), SLOT(close()) );
00016     menuBar()->addMenu( "File" )->addAction( a );
00017 }
00018
00019 trademgen::~~trademgen()
00020 {}
00021
00022 #include "trademgen.moc"
```

25.59 trademgen/batches/trademgen_with_db.cpp File Reference

```
#include <cassert>
#include <iostream>
#include <sstream>
#include <fstream>
#include <vector>
#include <string>
#include <boost/date_time/posix_time/posix_time.hpp>
#include <boost/date_time/gregorian/gregorian.hpp>
#include <boost/tokenizer.hpp>
#include <boost/program_options.hpp>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/basic/BasDBParams.hpp>
#include <stdair/basic/BasLogParams.hpp>
#include <trademgen/TRADEMGEN_Service.hpp>
#include <trademgen/config/trademgen-paths.hpp>
```

Typedefs

- `typedef std::vector< std::string > WordList_T`

Functions

- const std::string [K_TRADEMGEN_DEFAULT_LOG_FILENAME](#) ("trademgen_with_db.log")
- const std::string [K_TRADEMGEN_DEFAULT_INPUT_FILENAME](#) (STDAIR_SAMPLE_DIR"/demand01.-csv")
- const std::string [K_TRADEMGEN_DEFAULT_QUERY_STRING](#) ("my good old query")
- const std::string [K_TRADEMGEN_DEFAULT_DB_USER](#) ("dsim")
- const std::string [K_TRADEMGEN_DEFAULT_DB_PASSWD](#) ("dsim")
- const std::string [K_TRADEMGEN_DEFAULT_DB_DBNAME](#) ("sim_dsim")
- const std::string [K_TRADEMGEN_DEFAULT_DB_HOST](#) ("localhost")
- const std::string [K_TRADEMGEN_DEFAULT_DB_PORT](#) ("3306")
- void [tokeniseStringIntoWordList](#) (const std::string &iPhrase, [WordList_T](#) &ioWordList)
- std::string [createStringFromWordList](#) (const [WordList_T](#) &iWordList)
- template<class T >
std::ostream & [operator<<](#) (std::ostream &os, const std::vector< T > &v)
- int [readConfiguration](#) (int argc, char *argv[], bool &iolsBuiltin, stdair::RandomSeed_T &ioRandomSeed, std::string &ioQueryString, stdair::Filename_T &ioInputFilename, std::string &ioLogFilename, std::string &ioDBUser, std::string &ioDBPasswd, std::string &ioDBHost, std::string &ioDBPort, std::string &ioDBDBName)
- int [main](#) (int argc, char *argv[])

Variables

- const bool [K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT](#) = false
- const stdair::RandomSeed_T [K_TRADEMGEN_DEFAULT_RANDOM_SEED](#)
- const int [K_TRADEMGEN_EARLY_RETURN_STATUS](#) = 99

25.59.1 Typedef Documentation

25.59.1.1 typedef std::vector<std::string> WordList_T

Definition at line 24 of file [trademgen_with_db.cpp](#).

25.59.2 Function Documentation

25.59.2.1 const std::string K_TRADEMGEN_DEFAULT_LOG_FILENAME ("trademgen_with_db.log")

Default name and location for the log file.

25.59.2.2 const std::string K_TRADEMGEN_DEFAULT_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/demand01.-csv")

Default name and location for the (CSV) input file.

25.59.2.3 const std::string K_TRADEMGEN_DEFAULT_QUERY_STRING ("my good old query")

Default query string.

Referenced by [readConfiguration\(\)](#).

25.59.2.4 const std::string K_TRADEMGEN_DEFAULT_DB_USER ("dsim")

Default parameters for the database connection.

Referenced by [readConfiguration\(\)](#).

25.59.2.5 const std::string K_TRADEMGEN_DEFAULT_DB_PASSWD ("dsim")

Referenced by [readConfiguration\(\)](#).

25.59.2.6 `const std::string K_TRADEMGEN_DEFAULT_DB_DBNAME ("sim.dsim")`

Referenced by [readConfiguration\(\)](#).

25.59.2.7 `const std::string K_TRADEMGEN_DEFAULT_DB_HOST ("localhost")`

Referenced by [readConfiguration\(\)](#).

25.59.2.8 `const std::string K_TRADEMGEN_DEFAULT_DB_PORT ("3306")`

Referenced by [readConfiguration\(\)](#).

25.59.2.9 `void tokeniseStringIntoWordList (const std::string & iPhrase, WordList_T & ioWordList)`

Definition at line 67 of file [trademgen_with_db.cpp](#).

Referenced by [readConfiguration\(\)](#).

25.59.2.10 `std::string createStringFromWordList (const WordList_T & iWordList)`

Definition at line 89 of file [trademgen_with_db.cpp](#).

Referenced by [readConfiguration\(\)](#).

25.59.2.11 `template<class T> std::ostream& operator<< (std::ostream & os, const std::vector< T > & v)`

Definition at line 108 of file [trademgen_with_db.cpp](#).

25.59.2.12 `int readConfiguration (int argc, char * argv[], bool & iolsBuiltin, stdair::RandomSeed_T & ioRandomSeed, std::string & ioQueryString, stdair::Filename_T & ioInputFilename, std::string & ioLogFilename, std::string & ioDBUser, std::string & ioDBPasswd, std::string & ioDBHost, std::string & ioDBPort, std::string & ioDBDBName)`

Read and parse the command line options.

Definition at line 118 of file [trademgen_with_db.cpp](#).

References [createStringFromWordList\(\)](#), [K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT](#), [K_TRADEMGEN_DEFAULT_DB_DBNAME\(\)](#), [K_TRADEMGEN_DEFAULT_DB_HOST\(\)](#), [K_TRADEMGEN_DEFAULT_DB_PASSWD\(\)](#), [K_TRADEMGEN_DEFAULT_DB_PORT\(\)](#), [K_TRADEMGEN_DEFAULT_DB_USER\(\)](#), [K_TRADEMGEN_DEFAULT_INPUT_FILENAME\(\)](#), [K_TRADEMGEN_DEFAULT_LOG_FILENAME\(\)](#), [K_TRADEMGEN_DEFAULT_QUERY_STRING\(\)](#), [K_TRADEMGEN_DEFAULT_RANDOM_SEED](#), [K_TRADEMGEN_EARLY_RETURN_STATUS](#), [PACKAGE_NAME](#), [PACKAGE_VERSION](#), [PREFIXDIR](#), and [tokeniseStringIntoWordList\(\)](#).

25.59.2.13 `int main (int argc, char * argv[])`

Definition at line 288 of file [trademgen_with_db.cpp](#).

References [TRADEMGEN::TRADEMGEN_Service::buildSampleBom\(\)](#), [TRADEMGEN::TRADEMGEN_Service::displayAirlineListFromDB\(\)](#), [K_TRADEMGEN_EARLY_RETURN_STATUS](#), [TRADEMGEN::TRADEMGEN_Service::parseAndLoad\(\)](#), and [readConfiguration\(\)](#).

25.59.3 Variable Documentation

25.59.3.1 `const bool K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT = false`

Default for the input type. It can be either built-in or provided by an input file. That latter must then be given with the -i option.

Definition at line 43 of file [trademgen_with_db.cpp](#).

25.59.3.2 `const stdair::RandomSeed_T K_TRADEMGEN_DEFAULT_RANDOM_SEED`

Initial value:


```
stdair::DEFAULT_RANDOM_SEED
```

Default random generation seed (e.g., 120765987).

Definition at line 48 of file [trademgen_with_db.cpp](#).

25.59.3.3 const int K_TRADEMGEN_EARLY_RETURN_STATUS = 99

Early return status (so that it can be differentiated from an error).

Definition at line 115 of file [trademgen_with_db.cpp](#).

25.60 trademgen_with_db.cpp

```
00001 // STL
00002 #include <cassert>
00003 #include <iostream>
00004 #include <sstream>
00005 #include <fstream>
00006 #include <vector>
00007 #include <string>
00008 // Boost (Extended STL)
00009 #include <boost/date_time/posix_time/posix_time.hpp>
00010 #include <boost/date_time/gregorian/gregorian.hpp>
00011 #include <boost/tokenizer.hpp>
00012 #include <boost/program_options.hpp>
00013 // StdAir
00014 #include <stdair/stdair_basic_types.hpp>
00015 #include <stdair/basic/BasConst_General.hpp>
00016 #include <stdair/basic/BasDBParams.hpp>
00017 #include <stdair/basic/BasLogParams.hpp>
00018 // TraDemGen
00019 #include <trademgen/TRADEMGEN_Service.hpp>
00020 #include <trademgen/config/trademgen-paths.hpp>
00021
00022
00023 // ////////// Type definitions //////////
00024 typedef std::vector<std::string> WordList_T;
00025
00026
00027 // ////////// Constants //////////
00031 const std::string K_TRADEMGEN_DEFAULT_LOG_FILENAME ("trademgen_with_db.log");
00032
00036 const std::string K_TRADEMGEN_DEFAULT_INPUT_FILENAME (STDAIR_SAMPLE_DIR
00037                                                         "/demand01.csv");
00038
00043 const bool K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT = false;
00044
00048 const stdair::RandomSeed_T K_TRADEMGEN_DEFAULT_RANDOM_SEED =
00049     stdair::DEFAULT_RANDOM_SEED;
00050
00054 const std::string K_TRADEMGEN_DEFAULT_QUERY_STRING ("my good old query");
00055
00059 const std::string K_TRADEMGEN_DEFAULT_DB_USER ("dsim");
00060 const std::string K_TRADEMGEN_DEFAULT_DB_PASSWD ("dsim");
00061 const std::string K_TRADEMGEN_DEFAULT_DB_DBNAME ("sim_dsim");
00062 const std::string K_TRADEMGEN_DEFAULT_DB_HOST ("localhost");
00063 const std::string K_TRADEMGEN_DEFAULT_DB_PORT ("3306");
00064
00065
00066 // //////////////////////////////////////
00067 void tokeniseStringIntoWordList (const std::string& iPhrase,
00068                                 WordList_T& ioWordList) {
00069     // Empty the word list
00070     ioWordList.clear();
00071
00072     // Boost Tokeniser
00073     typedef boost::tokenizer<boost::char_separator<char> > Tokeniser_T;
00074
00075     // Define the separators
00076     const boost::char_separator<char> lSeparatorList ("
00077     .,:;|+-*/_!=@#$$%^&(){}[]?'\<>\"");
00077
00078     // Initialise the phrase to be tokenised
00079     Tokeniser_T lTokens (iPhrase, lSeparatorList);
00080     for (Tokeniser_T::const_iterator tok_iter = lTokens.begin();
00081          tok_iter != lTokens.end(); ++tok_iter) {
00082         const std::string& lTerm = *tok_iter;
00083         ioWordList.push_back (lTerm);
00084     }
```

```

00085
00086 }
00087
00088 // //////////////////////////////////////
00089 std::string createStringFromWordList (const WordList_T& iWordList) {
00090     std::ostringstream oStr;
00091
00092     unsigned short idx = iWordList.size();
00093     for (WordList_T::const_iterator itWord = iWordList.begin();
00094          itWord != iWordList.end(); ++itWord, --idx) {
00095         const std::string& lWord = *itWord;
00096         oStr << lWord;
00097         if (idx > 1) {
00098             oStr << " ";
00099         }
00100     }
00101
00102     return oStr.str();
00103 }
00104
00105
00106 // /////////// Parsing of Options & Configuration ///////////
00107 // A helper function to simplify the main part.
00108 template<class T> std::ostream& operator<< (std::ostream& os,
00109                                           const std::vector<T>& v) {
00110     std::copy (v.begin(), v.end(), std::ostream_iterator<T> (std::cout, " "));
00111     return os;
00112 }
00113
00115 const int K_TRADEMGEN_EARLY_RETURN_STATUS = 99;
00116
00118 int readConfiguration (int argc, char* argv[], bool& ioIsBuiltin,
00119                       stdair::RandomSeed_T& ioRandomSeed,
00120                       std::string& ioQueryString,
00121                       stdair::Filename_T& ioInputFilename,
00122                       std::string& ioLogFilename,
00123                       std::string& ioDBUser, std::string& ioDBPasswd,
00124                       std::string& ioDBHost, std::string& ioDBPort,
00125                       std::string& ioDBDBName) {
00126
00127     // Default for the built-in input
00128     ioIsBuiltin = K_TRADEMGEN_DEFAULT_BUILT_IN_INPUT;
00129
00130     // Initialise the travel query string, if that one is empty
00131     if (ioQueryString.empty() == true) {
00132         ioQueryString = K_TRADEMGEN_DEFAULT_QUERY_STRING;
00133     }
00134
00135     // Transform the query string into a list of words (STL strings)
00136     WordList_T lWordList;
00137     tokeniseStringIntoWordList (ioQueryString, lWordList);
00138
00139     // Declare a group of options that will be allowed only on command line
00140     boost::program_options::options_description generic ("Generic options");
00141     generic.add_options()
00142         ("prefix", "print installation prefix")
00143         ("version,v", "print version string")
00144         ("help,h", "produce help message");
00145
00146     // Declare a group of options that will be allowed both on command
00147     // line and in config file
00148     boost::program_options::options_description config ("Configuration");
00149     config.add_options()
00150         ("builtin,b",
00151          "The sample BOM tree can be either built-in or parsed from an input file.
00152          That latter must then be given with the -i/--input option")
00153         ("seed,s",
00154          boost::program_options::value<stdair::RandomSeed_T>(&ioRandomSeed)->
00155          default_value(K_TRADEMGEN_DEFAULT_RANDOM_SEED),
00156          "Seed for the random generation")
00157         ("input,i",
00158          boost::program_options::value< std::string >(&ioInputFilename)->
00159          default_value(K_TRADEMGEN_DEFAULT_INPUT_FILENAME),
00160          "(CVS) input file for the demand distributions")
00161         ("log,l",
00162          boost::program_options::value< std::string >(&ioLogFilename)->
00163          default_value(K_TRADEMGEN_DEFAULT_LOG_FILENAME),
00164          "Filepath for the logs")
00165         ("user,u",
00166          boost::program_options::value< std::string >(&ioDBUser)->default_value(
00167          K_TRADEMGEN_DEFAULT_DB_USER),
00168          "SQL database user (e.g., dsim)")
00169         ("passwd,p",
00170          boost::program_options::value< std::string >(&ioDBPasswd)->default_value(
00171          K_TRADEMGEN_DEFAULT_DB_PASSWD),
00172          "SQL database password (e.g., dsim)")
00173         ("host,H",

```

```

00168     boost::program_options::value< std::string >(&ioDBHost)->default_value(
00169         K_TRADEMGEN_DEFAULT_DB_HOST),
00170     ("port,p",
00171     boost::program_options::value< std::string >(&ioDBPort)->default_value(
00172         K_TRADEMGEN_DEFAULT_DB_PORT),
00173     "SQL database port (e.g., 3306)")
00174     boost::program_options::value< std::string >(&ioDBDBName)->default_value(
00175         K_TRADEMGEN_DEFAULT_DB_DBNAME),
00176     "SQL database name (e.g., sim_dsim)")
00177     ("query,q",
00178     boost::program_options::value< WordList_T >(&lWordList)->multitoken(),
00179     "Query word list")
00180 ;
00181 // Hidden options, will be allowed both on command line and
00182 // in config file, but will not be shown to the user.
00183 boost::program_options::options_description hidden ("Hidden options");
00184 hidden.add_options()
00185     ("copyright",
00186     boost::program_options::value< std::vector<std::string> >(),
00187     "Show the copyright (license)");
00188
00189 boost::program_options::options_description cmdline_options;
00190 cmdline_options.add(generic).add(config).add(hidden);
00191
00192 boost::program_options::options_description config_file_options;
00193 config_file_options.add(config).add(hidden);
00194
00195 boost::program_options::options_description visible ("Allowed options");
00196 visible.add(generic).add(config);
00197
00198 boost::program_options::positional_options_description p;
00199 p.add ("copyright", -1);
00200
00201 boost::program_options::variables_map vm;
00202 boost::program_options::
00203     store (boost::program_options::command_line_parser (argc, argv).
00204         options (cmdline_options).positional(p).run(), vm);
00205
00206 std::ifstream ifs ("trademgen_with_db.cfg");
00207 boost::program_options::store (parse_config_file (ifs, config_file_options),
00208     vm);
00209 boost::program_options::notify (vm);
00210
00211 if (vm.count ("help")) {
00212     std::cout << visible << std::endl;
00213     return K_TRADEMGEN_EARLY_RETURN_STATUS;
00214 }
00215
00216 if (vm.count ("version")) {
00217     std::cout << PACKAGE_NAME << ", version " << PACKAGE_VERSION << std::endl;
00218     return K_TRADEMGEN_EARLY_RETURN_STATUS;
00219 }
00220
00221 if (vm.count ("prefix")) {
00222     std::cout << "Installation prefix: " << PREFIXDIR << std::endl;
00223     return K_TRADEMGEN_EARLY_RETURN_STATUS;
00224 }
00225
00226 if (vm.count ("builtin")) {
00227     ioIsBuiltin = true;
00228 }
00229 const std::string isBuiltinStr = (ioIsBuiltin == true)?"yes":"no";
00230 std::cout << "The BOM should be built-in? " << isBuiltinStr << std::endl;
00231
00232 if (ioIsBuiltin == false) {
00233
00234     // The BOM tree should be built from parsing a demand input file
00235     if (vm.count ("input")) {
00236         ioInputFilename = vm["input"].as< std::string >();
00237         std::cout << "Input filename is: " << ioInputFilename << std::endl;
00238     } else {
00239         // The built-in option is not selected. However, no demand input file
00240         // is specified
00241         std::cerr << "Either one among the -b/--builtin and -i/--input "
00242             << "options must be specified" << std::endl;
00243     }
00244 }
00245
00246 if (vm.count ("log")) {
00247     ioLogFilename = vm["log"].as< std::string >();
00248     std::cout << "Log filename is: " << ioLogFilename << std::endl;
00249 }
00250
00251

```

```

00252     if (vm.count ("user")) {
00253         ioDBUser = vm["user"].as< std::string >();
00254         std::cout << "SQL database user name is: " << ioDBUser << std::endl;
00255     }
00256
00257     if (vm.count ("passwd")) {
00258         ioDBPasswd = vm["passwd"].as< std::string >();
00259         // std::cout << "SQL database user password is: " << ioDBPasswd <<
std::endl;
00260     }
00261
00262     if (vm.count ("host")) {
00263         ioDBHost = vm["host"].as< std::string >();
00264         std::cout << "SQL database host name is: " << ioDBHost << std::endl;
00265     }
00266
00267     if (vm.count ("port")) {
00268         ioDBPort = vm["port"].as< std::string >();
00269         std::cout << "SQL database port number is: " << ioDBPort << std::endl;
00270     }
00271
00272     if (vm.count ("dbname")) {
00273         ioDBDBName = vm["dbname"].as< std::string >();
00274         std::cout << "SQL database name is: " << ioDBDBName << std::endl;
00275     }
00276
00277     //
00278     std::cout << "The random generation seed is: " << ioRandomSeed << std::endl;
00279
00280     ioQueryString = createStringFromWordList (lWordList);
00281     std::cout << "The query string is: " << ioQueryString << std::endl;
00282
00283     return 0;
00284 }
00285
00286
00287 // ////////////////////////////////// M A I N //////////////////////////////////
00288 int main (int argc, char* argv[]) {
00289
00290     // State whether the BOM tree should be built-in or parsed from an input file
00291     bool isBuiltin;
00292
00293     // Random generation seed
00294     stdair::RandomSeed_T lRandomSeed;
00295
00296     // Query
00297     std::string lQuery;
00298
00299     // Input file name
00300     stdair::Filename_T lInputFilename;
00301
00302     // Output log File
00303     std::string lLogFilename;
00304
00305     // SQL database parameters
00306     std::string lDBUser;
00307     std::string lDBPasswd;
00308     std::string lDBHost;
00309     std::string lDBPort;
00310     std::string lDBDBName;
00311
00312     // Airline code
00313     const stdair::AirlineCode_T lAirlineCode ("BA");
00314
00315     // Call the command-line option parser
00316     const int lOptionParserStatus =
00317         readConfiguration (argc, argv, isBuiltin, lRandomSeed, lQuery,
00318             lInputFilename, lLogFilename,
00319             lDBUser, lDBPasswd, lDBHost, lDBPort, lDBDBName);
00320
00321     if (lOptionParserStatus == K_TRADEMGEN_EARLY_RETURN_STATUS) {
00322         return 0;
00323     }
00324
00325     // Set the database parameters
00326     stdair::BasDBParams lDBParams (lDBUser, lDBPasswd, lDBHost, lDBPort,
00327         lDBDBName);
00328
00329     // Set the log parameters
00330     std::ofstream logOutputFile;
00331     // open and clean the log outputfile
00332     logOutputFile.open (lLogFilename.c_str());
00333     logOutputFile.clear();
00334
00335     // Set up the log parameters
00336     const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
00337

```

```

00338 // Initialise the TraDemGen service object
00339 TRADEMGEN::TRADEMGEN_Service trademgenService (lLogParams, lDBParams,
00340                                                  lRandomSeed);
00341
00342 // Check whether or not a (CSV) input file should be read
00343 if (isBuiltin == true) {
00344     // Create a sample DemandStream object, and insert it within the BOM tree
00345     trademgenService.buildSampleBom();
00346 }
00347 } else {
00348     // Create the DemandStream objects, and insert them within the BOM tree
00349     trademgenService.parseAndLoad (lInputFilename);
00350 }
00351
00352 // Query the database
00353 trademgenService.displayAirlineListFromDB();
00354
00355 // Close the Log output file
00356 logOutputFile.close();
00357
00358 return 0;
00359 }

```

25.61 trademgen/bom/BomDisplay.cpp File Reference

```

#include <cassert>
#include <ostream>
#include <stdair/basic/BasConst_BomDisplay.hpp>
#include <stdair/bom/BomManager.hpp>
#include <stdair/bom/EventQueue.hpp>
#include <trademgen/bom/DemandStream.hpp>
#include <trademgen/bom/BomDisplay.hpp>

```

Classes

- struct [TRADEMGEN::FlagSaver](#)

Namespaces

- namespace [TRADEMGEN](#)

25.62 BomDisplay.cpp

```

00001 //////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
00002 // Import section
00003 //////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <ostream>
00007 // StdAir
00008 #include <stdair/basic/BasConst_BomDisplay.hpp>
00009 #include <stdair/bom/BomManager.hpp>
00010 #include <stdair/bom/EventQueue.hpp>
00011 // TraDemGen
00012 #include <trademgen/bom/DemandStream.hpp>
00013 #include <trademgen/bom/BomDisplay.hpp>
00014
00015 namespace TRADEMGEN {
00016
00022     struct FlagSaver {
00023     public:
00025         FlagSaver (std::ostream& oStream)
00026             : _oStream (oStream), _streamFlags (oStream.flags()) {
00027         }
00028
00030         ~FlagSaver() {
00031             // Reset formatting flags of the given output stream
00032             _oStream.flags (_streamFlags);
00033         }
00034

```

```

00035 private:
00037     std::ostream& _oStream;
00039     std::ios::fmtflags _streamFlags;
00040 };
00041
00042 // //////////////////////////////////////
00043 std::string BomDisplay::csvDisplay (const stdair::EventQueue& iEventQueue) {
00044     std::ostringstream oStream;
00045
00049     oStream << std::endl;
00050     oStream << "=====
"
00051         << std::endl;
00052     oStream << "EventQueue: " << iEventQueue.describeKey() << std::endl;
00053     oStream << "=====
"
00054         << std::endl;
00055
00056     // Check whether there are DemandStream objects
00057     if (stdair::BomManager::hasList<DemandStream> (iEventQueue) == false) {
00058         return oStream.str();
00059     }
00060
00061     // Retrieve the DemandStream list
00062     const DemandStreamList_T& lDemandStreamList =
00063         stdair::BomManager::getList<DemandStream> (iEventQueue);
00064
00065     // Browse the inventories
00066     for (DemandStreamList_T::const_iterator itDemandStream =
00067         lDemandStreamList.begin();
00068         itDemandStream != lDemandStreamList.end(); ++itDemandStream) {
00069         DemandStream* lDemandStream_ptr = *itDemandStream;
00070         assert (lDemandStream_ptr != NULL);
00071
00072         // Display the demand stream
00073         csvDisplay (oStream, *lDemandStream_ptr);
00074     }
00075
00076     return oStream.str();
00077 }
00078
00079 // //////////////////////////////////////
00080 void BomDisplay::csvDisplay (std::ostream& oStream,
00081                             const DemandStream& iDemandStream) {
00082     // Save the formatting flags for the given STL output stream
00083     FlagSaver flagSaver (oStream);
00084
00088     oStream << "+++++" << std::endl;
00089     oStream << iDemandStream.display();
00090     oStream << "+++++" << std::endl;
00091 }
00092
00093 }

```

25.63 trademgen/bom/BomDisplay.hpp File Reference

```

#include <iosfwd>
#include <string>

```

Classes

- class [TRADEMGEN::BomDisplay](#)
Utility class to display TraDemGen objects with a pretty format.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)

25.64 BomDisplay.hpp

```

00001 #ifndef __TRADEMGEN_BOM_BOMDISPLAY_HPP
00002 #define __TRADEMGEN_BOM_BOMDISPLAY_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <iosfwd>
00009 #include <string>
00010 // TraDemGen
00011
00013 namespace stdair {
00014     class EventQueue;
00015 }
00016
00017 namespace TRADEMGEN {
00018
00020     class DemandStream;
00021
00026     class BomDisplay {
00027     public:
00028         // ////////////////////////////////// Display support methods //////////////////////////////////
00037         static std::string csvDisplay (const stdair::EventQueue&);
00038
00047         static void csvDisplay (std::ostream&, const DemandStream&);
00048     };
00049
00050 }
00051 #endif // __TRADEMGEN_BOM_BOMDISPLAY_HPP

```

25.65 trademgen/bom/DemandStream.cpp File Reference

```

#include <cassert>
#include <sstream>
#include <cmath>
#include <iomanip>
#include <boost/make_shared.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/basic/BasConst_Inventory.hpp>
#include <stdair/basic/BasConst_Request.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/basic/BasConst_DemandGeneration.hpp>
#include <trademgen/bom/DemandStream.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.66 DemandStream.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 #include <cmath>
00008 #include <iomanip>
00009 // Boost
00010 #include <boost/make_shared.hpp>
00011 // StdAir
00012 #include <stdair/basic/BasConst_General.hpp>
00013 #include <stdair/basic/BasConst_Inventory.hpp>
00014 #include <stdair/basic/BasConst_Request.hpp>
00015 #include <stdair/bom/BookingRequestStruct.hpp>
00016 #include <stdair/service/Logger.hpp>
00017 // TraDemGen

```

```

00018 #include <trademgen/basic/BasConst_DemandGeneration.hpp>
00019 #include <trademgen/bom/DemandStream.hpp>
00020
00021 namespace TRADEMGEN {
00022
00023 // //////////////////////////////////////
00024 DemandStream::DemandStream()
00025 : _key (stdair::DEFAULT_ORIGIN, stdair::DEFAULT_DESTINATION,
00026        stdair::DEFAULT_DEPARTURE_DATE, stdair::DEFAULT_CABIN_CODE),
00027   _parent (NULL),
00028   _demandCharacteristics (ArrivalPatternCumulativeDistribution_T(),
00029                           POSProbabilityMassFunction_T(),
00030                           ChannelProbabilityMassFunction_T(),
00031                           TripTypeProbabilityMassFunction_T(),
00032                           StayDurationProbabilityMassFunction_T(),
00033                           FrequentFlyerProbabilityMassFunction_T(),
00034                           PreferredDepartureTimeContinuousDistribution_T(),
00035                           0.0,
00036                           ValueOfTimeContinuousDistribution_T()),
00037   _posProMass (DEFAULT_POS_PROBALILITY_MASS),
00038   _firstDateTimeRequest (true) {
00039     assert (false);
00040 }
00041
00042 // //////////////////////////////////////
00043 DemandStream::DemandStream (const DemandStream&)
00044 : _key (stdair::DEFAULT_ORIGIN, stdair::DEFAULT_DESTINATION,
00045        stdair::DEFAULT_DEPARTURE_DATE, stdair::DEFAULT_CABIN_CODE),
00046   _parent (NULL),
00047   _demandCharacteristics (ArrivalPatternCumulativeDistribution_T(),
00048                           POSProbabilityMassFunction_T(),
00049                           ChannelProbabilityMassFunction_T(),
00050                           TripTypeProbabilityMassFunction_T(),
00051                           StayDurationProbabilityMassFunction_T(),
00052                           FrequentFlyerProbabilityMassFunction_T(),
00053                           PreferredDepartureTimeContinuousDistribution_T(),
00054                           0.0,
00055                           ValueOfTimeContinuousDistribution_T()),
00056   _posProMass (DEFAULT_POS_PROBALILITY_MASS),
00057   _firstDateTimeRequest (true) {
00058     assert (false);
00059 }
00060
00061 // //////////////////////////////////////
00062 DemandStream::DemandStream (const Key_T& iKey) :
00063   _key (iKey) {
00064 }
00065
00066 // //////////////////////////////////////
00067 DemandStream::~DemandStream() {
00068 }
00069
00070 // //////////////////////////////////////
00071 std::string DemandStream::toString() const {
00072   std::ostringstream ostr;
00073   ostr << _key.toString();
00074   return ostr.str();
00075 }
00076
00077 // //////////////////////////////////////
00078 void DemandStream::
00079 setAll (const ArrivalPatternCumulativeDistribution_T& iArrivalPattern,
00080        const POSProbabilityMassFunction_T& iPOSProbMass,
00081        const ChannelProbabilityMassFunction_T& iChannelProbMass,
00082        const TripTypeProbabilityMassFunction_T& iTripTypeProbMass,
00083        const StayDurationProbabilityMassFunction_T& iStayDurationProbMass,
00084        const FrequentFlyerProbabilityMassFunction_T& iFrequentFlyerProbMass,
00085        const PreferredDepartureTimeContinuousDistribution_T&
00086 iPreferredDepartureTimeContinuousDistribution,
00087        const stdair::WTP_T& iMinWTP,
00088        const ValueOfTimeContinuousDistribution_T&
00089 iValueOfTimeContinuousDistribution,
00090        const DemandDistribution& iDemandDistribution,
00091        stdair::BaseGenerator_T& ioSharedGenerator,
00092        const stdair::RandomSeed_T& iRequestDateTimeSeed,
00093        const stdair::RandomSeed_T& iDemandCharacteristicsSeed,
00094        const POSProbabilityMass_T& iDefaultPOSProbabilityMass) {
00095   setDemandCharacteristics (iArrivalPattern, iPOSProbMass,
00096                             iChannelProbMass, iTripTypeProbMass,
00097                             iStayDurationProbMass, iFrequentFlyerProbMass,
00098                             iPreferredDepartureTimeContinuousDistribution,
00099                             iMinWTP, iValueOfTimeContinuousDistribution);
00100   setDemandDistribution (iDemandDistribution);
00101   setTotalNumberOfRequestsToBeGenerated (0);
00102   setRequestDateTimeRandomGeneratorSeed (iRequestDateTimeSeed);

```



```

00103     setDemandCharacteristicsRandomGeneratorSeed (iDemandCharacteristicsSeed);
00104     setPOSProbabilityMass (iDefaultPOSProbabilityMass);
00105
00106     //
00107     init (ioSharedGenerator);
00108 }
00109
00110 // //////////////////////////////////////
00111 std::string DemandStream::display() const {
00112     std::ostringstream oStr;
00113
00114     oStr << "Demand stream key: " << _key.toString() << std::endl;
00115
00116     //
00117     oStr << _demandCharacteristics.describe();
00118
00119     //
00120     oStr << _demandDistribution.describe() << " => "
00121         << _totalNumberOfRequestsToBeGenerated << " to be generated"
00122         << std::endl;
00123
00124     //
00125     oStr << "Random generation context: " << _randomGenerationContext
00126         << std::endl;
00127
00128     //
00129     oStr << "Random generator for date-time: "
00130         << _requestDateTimeRandomGenerator << std::endl;
00131     oStr << "Random generator for demand characteristics: "
00132         << _demandCharacteristicsRandomGenerator << std::endl;
00133
00134     //
00135     oStr << _posProMass.displayProbabilityMass() << std::endl;
00136
00137     return oStr.str();
00138 }
00139
00140 // //////////////////////////////////////
00141 void DemandStream::init (stdair::BaseGenerator_T& ioSharedGenerator) {
00142
00143     // Generate the number of requests
00144     const stdair::RealNumber_T lMu = _demandDistribution._meanNumberOfRequests;
00145     const stdair::RealNumber_T lSigma =
00146         _demandDistribution._stdDevNumberOfRequests;
00147
00148     stdair::NormalDistribution_T lDistrib (lMu, lSigma);
00149     stdair::NormalGenerator_T lNormalGen (ioSharedGenerator, lDistrib);
00150
00151     const stdair::RealNumber_T lRealNumberOfRequestsToBeGenerated = lNormalGen()
00152 ;
00153     const stdair::NbOfRequests_T lIntegerNumberOfRequestsToBeGenerated =
00154         std::floor (lRealNumberOfRequestsToBeGenerated + 0.5);
00155
00156     _totalNumberOfRequestsToBeGenerated = lIntegerNumberOfRequestsToBeGenerated
00157 ;
00158     _stillHavingRequestsToBeGenerated = true;
00159     _firstDateTimeRequest = true;
00160 }
00161
00162 // //////////////////////////////////////
00163 const bool DemandStream::
00164 stillHavingRequestsToBeGenerated (const stdair::DemandGenerationMethod&
iDemandGenerationMethod) const {
00165
00166     const stdair::DemandGenerationMethod::EN_DemandGenerationMethod&
lENDemandGenerationMethod =
00167         iDemandGenerationMethod.getMethod();
00168     if (lENDemandGenerationMethod == stdair::DemandGenerationMethod::STA_ORD) {
00169         bool hasStillHavingRequestsToBeGenerated = true;
00170
00171         // Check whether enough requests have already been generated
00172         const stdair::Count_T lNbOfRequestsGeneratedSoFar =
00173             _randomGenerationContext.getNumberOfRequestsGeneratedSoFar();
00174
00175         const stdair::Count_T lRemainingNumberOfRequestsToBeGenerated =
00176             _totalNumberOfRequestsToBeGenerated - lNbOfRequestsGeneratedSoFar;
00177
00178         if (lRemainingNumberOfRequestsToBeGenerated <= 0) {
00179             hasStillHavingRequestsToBeGenerated = false;
00180         }
00181
00182         return hasStillHavingRequestsToBeGenerated;
00183     } else {
00184         return _stillHavingRequestsToBeGenerated;
00185     }

```

```

00186     }
00187
00188     // //////////////////////////////////////
00189     const stdair::DateTime_T DemandStream::generateTimeOfRequestPoissonProcess()
00190     {
00191         // Prepare arrival pattern.
00192         const ContinuousFloatDuration_T& lArrivalPattern =
00193             _demandCharacteristics._arrivalPattern;
00194
00195         const stdair::Time_T lHardcodedReferenceDepartureTime =
00196             boost::posix_time::hours (8);
00197
00198         // Prepare departure date time.
00199         const stdair::DateTime_T lDepartureDateTime =
00200             boost::posix_time::ptime (_key.getPreferredDepartureDate(),
00201                                     lHardcodedReferenceDepartureTime);
00202
00203         // If no request has been generated so far...
00204         if (_firstDateTimeRequest) {
00205             const stdair::Probability_T lProbabilityFirstRequest = 0;
00206
00207             // Get the lower bound of the arrival pattern (corresponding
00208             // to a cumulative probability of 0).
00209             _dateTimeLastRequest =
00210                 lArrivalPattern.getValue (lProbabilityFirstRequest);
00211
00212             _firstDateTimeRequest = false;
00213         }
00214
00215         // Sanity check.
00216         assert (_firstDateTimeRequest == false);
00217
00218         // If the date time of the last request is equal to the lower bound of
00219         // the last daily rate interval (default value is -1, meaning one day
00220         // before departure), we stopped generating request by returning a
00221         // request date time after departure date time.
00222         if (_dateTimeLastRequest == DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN) {
00223             _stillHavingRequestsToBeGenerated = false;
00224
00225             // Get a positive number of days.
00226             const stdair::Duration_T lDifferenceBetweenDepartureAndThisLowerBound =
00227                 convertFloatIntoDuration (-DEFAULT_LAST_LOWER_BOUND_ARRIVAL_PATTERN);
00228
00229             // Calculate a request date-time after the departure date time to end
00230             // the demand generation algorithm.
00231             const stdair::DateTime_T oDateTimeThisRequest =
00232                 lDepartureDateTime + lDifferenceBetweenDepartureAndThisLowerBound;
00233
00234             return oDateTimeThisRequest;
00235         }
00236
00237         // Get the upper bound of the current daily rate interval.
00238         stdair::FloatDuration_T lUpperBound =
00239             lArrivalPattern.getUpperBound (_dateTimeLastRequest);
00240
00241         // Compute the daily rate demand.
00242         double lDailyRate = lArrivalPattern.getDerivativeValue(_dateTimeLastRequest);
00243
00244         // Get the expected average number of requests.
00245         const double lDemandMean = _demandDistribution._meanNumberOfRequests;
00246         // Multiply the daily rate by the expected average number of requests.
00247         lDailyRate *= lDemandMean;
00248
00249         // Generate an exponential variable.
00250         const stdair::FloatDuration_T lExponentialVariable =
00251             _requestDateTimeRandomGenerator.generateExponential (lDailyRate);
00252
00253         // Compute the new date time request.
00254         const stdair::FloatDuration_T lDateTimeThisRequest =
00255             _dateTimeLastRequest + lExponentialVariable;
00256
00257         stdair::DateTime_T oDateTimeThisRequest;
00258
00259         // Verify if this request is in the given daily rate interval.
00260         if (lDateTimeThisRequest < lUpperBound) {
00261             // Conversion.
00262             const stdair::Duration_T lDifferenceBetweenDepartureAndThisRequest =
00263                 convertFloatIntoDuration (lDateTimeThisRequest);
00264
00265             // The request date-time is derived from departure date and arrival
00266             pattern.
00267             oDateTimeThisRequest = lDepartureDateTime
00268                 + lDifferenceBetweenDepartureAndThisRequest;
00269
00270             // Remember this date time request.

```

```

00270     _dateTimeLastRequest = lDateTimeThisRequest;
00271
00272     // Update the counter of requests generated so far.
00273     incrementGeneratedRequestsCounter();
00274
00275     const double lRefDateTimeThisRequest = lDateTimeThisRequest + double(2880
0.001/86400.0);
00276     STDAIR_LOG_NOTIFICATION (boost::gregorian::to_iso_string(_key.
getPreferredDepartureDate()) << " " << std::setprecision(10) <<
lRefDateTimeThisRequest);
00277 } else {
00278
00279     // The current request is not in the given daily rate interval.
00280     // Change the daily rate.
00281     _dateTimeLastRequest = lUpperBound;
00282
00283     // Generate a date time request in the new daily rate interval.
00284     oDateTimeThisRequest = generateTimeOfRequestPoissonProcess ();
00285 }
00286
00287 return oDateTimeThisRequest;
00288 }
00289
00290 // //////////////////////////////////////
00291 const stdair::DateTime_T DemandStream::generateTimeOfRequestStatisticsOrder()
{
00292
00309 //
00310 // Calculate the result of the formula above step by step.
00311 //
00312
00313 // 1) Get the number of requests generated so far.
00314 // (equal to k - 1)
00315 const stdair::Count_T& lNbOfRequestsGeneratedSoFar =
00316     _randomGenerationContext.getNumberOfRequestsGeneratedSoFar();
00317
00318 // 2) Deduce the number of requests not generated yet.
00319 // (equal to n - k + 1)
00320 const stdair::Count_T lRemainingNumberOfRequestsToBeGenerated =
00321     _totalNumberOfRequestsToBeGenerated - lNbOfRequestsGeneratedSoFar;
00322
00323 // Assert that there are still requests to be generated.
00324 assert (lRemainingNumberOfRequestsToBeGenerated > 0);
00325
00326 // 3) Inverse the number of requests not generated yet.
00327 // 1/(n - k + 1)
00328 const double lRemainingRate =
00329     1.0 / static_cast<double> (lRemainingNumberOfRequestsToBeGenerated);
00330
00331 // 4) Get the cumulative probability so far and take its complement.
00332 // (equal to 1 - x(k-1))
00333 const stdair::Probability_T& lCumulativeProbabilitySoFar =
00334     _randomGenerationContext.getCumulativeProbabilitySoFar();
00335 const stdair::Probability_T lComplementOfCumulativeProbabilitySoFar =
00336     1.0 - lCumulativeProbabilitySoFar;
00337
00338 // 5) Draw a random variable y and calculate the factor equal to
00339 // (1 - y)^(1/(n - k + 1)).
00340 const stdair::Probability_T& lVariate = _requestDateTimeRandomGenerator();
00341 double lFactor = std::pow (1.0 - lVariate, lRemainingRate);
00342 if (lFactor >= 1.0 - 1e-6){
00343     lFactor = 1.0 - 1e-6;
00344 }
00345
00346 // 6) Apply the whole formula above to calculate the cumulative probability
00347 // of the new request.
00348 // (equal to 1 - (1 - x(k-1)) (1 - y)^(1/(n - k + 1)))
00349 const stdair::Probability_T lCumulativeProbabilityThisRequest =
00350     1.0 - lComplementOfCumulativeProbabilitySoFar * lFactor;
00351
00352 // Now that the cumulative proportion of events generated has been
00353 // calculated, we deduce from the arrival pattern the arrival time of the
00354 // k-th event.
00355 const stdair::FloatDuration_T lNumberOfDaysBetweenDepartureAndThisRequest =
00356     _demandCharacteristics._arrivalPattern.getValue (
lCumulativeProbabilityThisRequest);
00357
00358 const stdair::Duration_T lDifferenceBetweenDepartureAndThisRequest =
00359     convertFloatIntoDuration (lNumberOfDaysBetweenDepartureAndThisRequest);
00360
00361 const stdair::Time_T lHardcodedReferenceDepartureTime =
00362     boost::posix_time::hours (8);
00363
00364 const stdair::DateTime_T lDepartureDateTime =
00365     boost::posix_time::ptime (_key.getPreferredDepartureDate(),
lHardcodedReferenceDepartureTime);
00366
00367

```

```

00368     // The request date-time is derived from departure date and arrival
pattern.
00369     const stdair::DateTime_T oDateTimeThisRequest =
00370         lDepartureDateTime + lDifferenceBetweenDepartureAndThisRequest;
00371
00372     // Update random generation context
00373     _randomGenerationContext.setCumulativeProbabilitySoFar (
lCumulativeProbabilityThisRequest);
00374
00375     // Update the counter of requests generated so far.
00376     incrementGeneratedRequestsCounter();
00377
00378     // DEBUG
00379     // STDAIR_LOG_DEBUG (lCumulativeProbabilityThisRequest << "; "
00380     //                  << lNumberOfDaysBetweenDepartureAndThisRequest);
00381
00382     // NOTIFICATION
00383     double lRefNumberOfDaysBetweenDepartureAndThisRequest =
00384         lNumberOfDaysBetweenDepartureAndThisRequest + double(1.0/3.0);
00385     STDAIR_LOG_NOTIFICATION (boost::gregorian::to_iso_string(_key.
getPreferredDepartureDate()) << ";" << std::setprecision(10) <<
lRefNumberOfDaysBetweenDepartureAndThisRequest);
00386
00387     return oDateTimeThisRequest;
00388 }
00389
00390 // //////////////////////////////////////
00391
00392 const stdair::Duration_T DemandStream:
00393 convertFloatIntoDuration (const stdair::FloatDuration_T iNumberOfDays) {
00394
00395     // Convert the number of days in number of seconds + number of milliseconds
00396     const stdair::FloatDuration_T lNumberOfSeconds =
00397         iNumberOfDays * stdair::SECONDS_IN_ONE_DAY;
00398
00399     // Get the number of seconds.
00400     const stdair::IntDuration_T lIntNumberOfSeconds =
00401         std::floor (lNumberOfSeconds);
00402
00403     // Get the number of milliseconds.
00404     const stdair::FloatDuration_T lNumberOfMilliseconds =
00405         (lNumberOfSeconds - lIntNumberOfSeconds)
00406         * stdair::MILLISECONDS_IN_ONE_SECOND;
00407
00408     // +1 is a trick to ensure that the next Event is strictly later
00409     // than the current one
00410     const stdair::IntDuration_T lIntNumberOfMilliseconds =
00411         std::floor (lNumberOfMilliseconds) + 1;
00412
00413     // Convert the number of seconds and milliseconds into a duration.
00414     const stdair::Duration_T lDifferenceBetweenDepartureAndThisRequest =
00415         boost::posix_time::seconds (lIntNumberOfSeconds)
00416         + boost::posix_time::millisec (lIntNumberOfMilliseconds);
00417
00418     return lDifferenceBetweenDepartureAndThisRequest;
00419 }
00420
00421 // //////////////////////////////////////
00422 const stdair::AirportCode_T DemandStream::generatePOS() {
00423
00424     // Generate a random number between 0 and 1.
00425     const stdair::Probability_T& lVariate =
_demandCharacteristicsRandomGenerator();
00426     const stdair::AirportCode_T& oPOS = _demandCharacteristics.generatePOSValue (
lVariate);
00427
00428     return oPOS;
00429 }
00430
00431 // //////////////////////////////////////
00432 const stdair::ChannelLabel_T DemandStream::generateChannel() {
00433     // Generate a random number between 0 and 1.
00434     const stdair::Probability_T lVariate =
_demandCharacteristicsRandomGenerator();
00435
00436     return _demandCharacteristics._channelProbabilityMass.getValue (lVariate);
00437 }
00438
00439 // //////////////////////////////////////
00440 const stdair::TripType_T DemandStream::generateTripType() {
00441     // Generate a random number between 0 and 1.
00442     const stdair::Probability_T lVariate =
_demandCharacteristicsRandomGenerator();
00443
00444     return _demandCharacteristics._tripTypeProbabilityMass.getValue (lVariate);
00445 }
00446
00447 }
00448

```

```

00449 // //////////////////////////////////////
00450 const stdair::DayDuration_T DemandStream::generateStayDuration() {
00451     // Generate a random number between 0 and 1.
00452     const stdair::Probability_T lVariate =
00453         _demandCharacteristicsRandomGenerator();
00454
00455     return _demandCharacteristics._stayDurationProbabilityMass.getValue (
00456         lVariate);
00457 }
00458 // //////////////////////////////////////
00459 const stdair::FrequentFlyer_T DemandStream::generateFrequentFlyer() {
00460     // Generate a random number between 0 and 1.
00461     const stdair::Probability_T lVariate =
00462         _demandCharacteristicsRandomGenerator();
00463
00464     return _demandCharacteristics._frequentFlyerProbabilityMass.getValue (
00465         lVariate);
00466 }
00467 // //////////////////////////////////////
00468 const stdair::Duration_T DemandStream::generatePreferredDepartureTime() {
00469     // Generate a random number between 0 and 1.
00470     const stdair::Probability_T lVariate =
00471         _demandCharacteristicsRandomGenerator();
00472     const stdair::IntDuration_T lNbOfSeconds = _demandCharacteristics.
00473         _preferredDepartureTimeCumulativeDistribution.getValue (lVariate);
00474
00475     const stdair::Duration_T oTime = boost::posix_time::seconds (lNbOfSeconds);
00476
00477     return oTime;
00478 }
00479 // //////////////////////////////////////
00480 const stdair::WTP_T DemandStream::
00481 generateWTP (stdair::RandomGeneration& ioGenerator,
00482             const stdair::Date_T& iDepartureDate,
00483             const stdair::DateTime_T& iDateTimeThisRequest,
00484             const stdair::DayDuration_T& iDurationOfStay) {
00485     const stdair::Date_T lDateThisRequest = iDateTimeThisRequest.date();
00486     const stdair::DateOffset_T lAP = iDepartureDate - lDateThisRequest;
00487     const stdair::DayDuration_T lAPInDays = lAP.days();
00488
00489     stdair::RealNumber_T lProb = -lAPInDays;
00490     // 1 - lAPInDays / DEFAULT_MAX_ADVANCE_PURCHASE;
00491     if (lProb < 0.0) { lProb = 0.0; }
00492     stdair::RealNumber_T lFrat5Coef =
00493         _demandCharacteristics._frat5Pattern.getValue (lProb);
00494
00495     const stdair::WTP_T lWTP = _demandCharacteristics._minWTP
00496         * (1.0 + (lFrat5Coef - 1.0) * log(ioGenerator()) / log(0.5));
00497
00498     return lWTP;
00499 }
00500 // //////////////////////////////////////
00501 const stdair::PriceValue_T DemandStream::generateValueOfTime() {
00502     // Generate a random number between 0 and 1.
00503     const stdair::Probability_T lVariate =
00504         _demandCharacteristicsRandomGenerator();
00505
00506     return _demandCharacteristics._valueOfTimeCumulativeDistribution.getValue (
00507         lVariate);
00508 }
00509 // //////////////////////////////////////
00510 stdair::BookingRequestPtr_T DemandStream::
00511 generateNextRequest (stdair::RandomGeneration& ioGenerator,
00512                     const stdair::DemandGenerationMethod&
00513                     iDemandGenerationMethod) {
00514     // Origin
00515     const stdair::AirportCode_T& lOrigin = _key.getOrigin();
00516     // Destination
00517     const stdair::AirportCode_T& lDestination = _key.getDestination();
00518     // Preferred departure date
00519     const stdair::Date_T& lPreferredDepartureDate =
00520         _key.getPreferredDepartureDate();
00521     // Preferred cabin
00522     const stdair::CabinCode_T& lPreferredCabin = _key.getPreferredCabin();
00523     // Party size
00524     const stdair::NbOfSeats_T lPartySize = stdair::DEFAULT_PARTY_SIZE;
00525     // POS
00526     const stdair::AirportCode_T lPOS = generatePOS();
00527
00528     // Compute the request date time with the correct algorithm.
00529     stdair::DateTime_T lDateTimeThisRequest;

```

```

00532     const stdair::DemandGenerationMethod::EN_DemandGenerationMethod&
1ENDemandGenerationMethod =
00533         iDemandGenerationMethod.getMethod();
00534     switch(1ENDemandGenerationMethod) {
00535     case stdair::DemandGenerationMethod::POI_PRO:
00536         lDateTimeThisRequest = generateTimeOfRequestPoissonProcess(); break;
00537     case stdair::DemandGenerationMethod::STA_ORD:
00538         lDateTimeThisRequest = generateTimeOfRequestStatisticsOrder(); break;
00539     default: assert (false); break;
00540     }
00541
00542     // Booking channel.
00543     const stdair::ChannelLabel_T lChannelLabel = generateChannel();
00544     // Trip type.
00545     const stdair::TripType_T lTripType = generateTripType();
00546     // Stay duration.
00547     const stdair::DayDuration_T lStayDuration = generateStayDuration();
00548     // Frequet flyer type.
00549     const stdair::FrequentFlyer_T lFrequentFlyer = generateFrequentFlyer();
00550     // Preferred departure time.
00551     const stdair::Duration_T lPreferredDepartureTime =
00552         generatePreferredDepartureTime();
00553     // Value of time
00554     const stdair::PriceValue_T lValueOfTime = generateValueOfTime();
00555     // WTP
00556     const stdair::WTP_T lWTP = generateWTP (ioGenerator, lPreferredDepartureDate
00557
00558         ,
00559         lDateTimeThisRequest, lStayDuration)
00560
00561     ;
00562     // TODO 1: understand why the following form does not work, knowing
00563     // that:
00564     // typedef boost::shared_ptr<stdair::BookingRequestStruct>
00565     stdair::BookingRequestPtr_T
00566     // stdair::BookingRequestPtr_T oBookingRequest_ptr =
00567     // boost::make_shared<stdair::BookingRequestStruct> ();
00568
00569     // TODO 2: move the creation of the structure out of the BOM layer
00570     // (into the command layer, e.g., within the DemandManager command).
00571
00572     // Create the booking request
00573     stdair::BookingRequestPtr_T oBookingRequest_ptr =
00574         stdair::BookingRequestPtr_T
00575         (new stdair::BookingRequestStruct (describeKey(), lOrigin,
00576             lDestination, lPOS,
00577             lPreferredDepartureDate,
00578             lDateTimeThisRequest,
00579             lPreferredCabin, lPartySize,
00580             lChannelLabel, lTripType,
00581             lStayDuration, lFrequentFlyer,
00582             lPreferredDepartureTime,
00583             lWTP, lValueOfTime));
00584
00585     // DEBUG
00586     // STDAIR_LOG_DEBUG ("\\n[BKG] " << oBookingRequest_ptr->describe());
00587
00588     return oBookingRequest_ptr;
00589 }
00590
00591 ///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
00592 void DemandStream::reset (stdair::BaseGenerator_T& ioSharedGenerator) {
00593     _randomGenerationContext.reset();
00594     init (ioSharedGenerator);
00595 }
00596
00597 }
```

25.67 trademgen/bom/DemandStream.hpp File Reference

```

#include <stdair/bom/BomAbstract.hpp>
#include <stdair/bom/BookingRequestTypes.hpp>
#include <stdair/basic/RandomGeneration.hpp>
#include <stdair/basic/DemandGenerationMethod.hpp>
#include <trademgen/basic/DemandCharacteristics.hpp>
#include <trademgen/basic/DemandDistribution.hpp>
#include <trademgen/basic/RandomGenerationContext.hpp>
#include <trademgen/bom/DemandStreamKey.hpp>
#include <trademgen/bom/DemandStreamTypes.hpp>
```

Classes

- class [TRADEMGEN::DemandStream](#)
Class modeling a demand stream.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)

25.68 DemandStream.hpp

```

00001 #ifndef __TRADEMGEN_BOM_DEMANDSTREAM_HPP
00002 #define __TRADEMGEN_BOM_DEMANDSTREAM_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/bom/BomAbstract.hpp>
00009 #include <stdair/bom/BookingRequestTypes.hpp>
00010 #include <stdair/basic/RandomGeneration.hpp>
00011 #include <stdair/basic/DemandGenerationMethod.hpp>
00012 // TraDemGen
00013 #include <trademgen/basic/DemandCharacteristics.hpp>
00014 #include <trademgen/basic/DemandDistribution.hpp>
00015 #include <trademgen/basic/RandomGenerationContext.hpp>
00016 #include <trademgen/bom/DemandStreamKey.hpp>
00017 #include <trademgen/bom/DemandStreamTypes.hpp>
00018
00020 namespace stdair {
00021     class FacBomManager;
00022     template <typename BOM> class FacBom;
00023 }
00024
00025 namespace TRADEMGEN {
00026
00030     class DemandStream : public stdair::BomAbstract {
00031     public:
00032         template <typename BOM> friend class stdair::FacBom;
00033         friend class stdair::FacBomManager;
00034
00035         // ////////////////////////////////// Type definitions //////////////////////////////////
00039         typedef DemandStreamKey Key_T;
00040
00041     public:
00042         // ////////////////////////////////// Getters //////////////////////////////////
00043         const Key_T& getKey() const {
00044             return _key;
00045         }
00046
00050         BomAbstract* const getParent() const {
00051             return _parent;
00052         }
00053
00055         const stdair::AirportCode_T& getOrigin() const {
00056             return _key.getOrigin();
00057         }
00058
00060         const stdair::AirportCode_T& getDestination() const {
00061             return _key.getDestination();
00062         }
00063
00065         const stdair::Date_T& getPreferredDepartureDate() const {
00066             return _key.getPreferredDepartureDate();
00067         }
00068
00070         const stdair::CabinCode_T& getPreferredCabin() const {
00071             return _key.getPreferredCabin();
00072         }
00073
00075         const stdair::HolderMap_T& getHolderMap() const {

```

```

00076         return _holderMap;
00077     }
00078
00080     const DemandCharacteristics& getDemandCharacteristics() const {
00081         return _demandCharacteristics;
00082     }
00083
00085     const DemandDistribution& getDemandDistribution() const {
00086         return _demandDistribution;
00087     }
00088
00090     const stdair::NbOfRequests_T& getTotalNumberOfRequestsToBeGenerated() const
00091     {
00092         return _totalNumberOfRequestsToBeGenerated;
00093     }
00095     const stdair::NbOfRequests_T& getMeanNumberOfRequests() const {
00096         return _demandDistribution._meanNumberOfRequests;
00097     }
00098
00100     const stdair::StdDevValue_T& getStdDevNumberOfRequests() const {
00101         return _demandDistribution._stdDevNumberOfRequests;
00102     }
00103
00105     const stdair::Count_T& getNumberOfRequestsGeneratedSoFar() const {
00106         return _randomGenerationContext.getNumberOfRequestsGeneratedSoFar();
00107     }
00108
00113     const POSProbabilityMass_T& getPOSProbabilityMass() const {
00114         return _posProMass;
00115     }
00116
00117
00118     public:
00119         // ////////////////////////////////// Setters //////////////////////////////////
00121         void setNumberOfRequestsGeneratedSoFar (const stdair::Count_T& iCount) {
00122             _randomGenerationContext.setNumberOfRequestsGeneratedSoFar (iCount);
00123         }
00124
00126         void setDemandDistribution (const DemandDistribution& iDemandDistribution)
00127         {
00128             _demandDistribution = iDemandDistribution;
00129         }
00131         void
00132         setDemandCharacteristics (const ArrivalPatternCumulativeDistribution_T&
00133 iArrivalPattern,
00134                                const POSProbabilityMassFunction_T& iPOSProbMass,
00135                                const ChannelProbabilityMassFunction_T&
00136 iChannelProbMass,
00137                                const TripTypeProbabilityMassFunction_T&
00138 iTripTypeProbMass,
00139                                const StayDurationProbabilityMassFunction_T&
00140 iStayDurationProbMass,
00141                                const FrequentFlyerProbabilityMassFunction_T&
00142 iFrequentFlyerProbMass,
00143                                const
00144 PreferredDepartureTimeContinuousDistribution_T&
00145 iPreferredDepartureTimeContinuousDistribution,
00146                                const stdair::WTP_T& iMinWTP,
00147                                const ValueOfTimeContinuousDistribution_T&
00148 iValueOfTimeContinuousDistribution) {
00149             _demandCharacteristics =
00150                 DemandCharacteristics (iArrivalPattern, iPOSProbMass,
00151 iChannelProbMass, iTripTypeProbMass,
00152 iStayDurationProbMass, iFrequentFlyerProbMass,
00153 iPreferredDepartureTimeContinuousDistribution,
00154 iMinWTP, iValueOfTimeContinuousDistribution);
00155         }
00156         void setTotalNumberOfRequestsToBeGenerated (const stdair::NbOfRequests_T&
00157 iNbOfRequests) {
00158             _totalNumberOfRequestsToBeGenerated = iNbOfRequests;
00159         }
00161         void setRequestDateTimeRandomGeneratorSeed (const stdair::RandomSeed_T&
00162 iSeed) {
00163             _requestDateTimeRandomGenerator.init (iSeed);
00164         }
00166         void setDemandCharacteristicsRandomGeneratorSeed (const
00167 stdair::RandomSeed_T& iSeed) {
00168             _demandCharacteristicsRandomGenerator.init (iSeed);
00169         }
00171         void setPOSProbabilityMass (const POSProbabilityMass_T& iProbMass) {
00172             _posProMass = iProbMass;
00173         }

```



```

00170     }
00171
00175     void setAll (const ArrivalPatternCumulativeDistribution_T&,
00176                 const POSProbabilityMassFunction_T&,
00177                 const ChannelProbabilityMassFunction_T&,
00178                 const TripTypeProbabilityMassFunction_T&,
00179                 const StayDurationProbabilityMassFunction_T&,
00180                 const FrequentFlyerProbabilityMassFunction_T&,
00181                 const PreferredDepartureTimeContinuousDistribution_T&,
00182                 const stdair::WTP_T&,
00183                 const ValueOfTimeContinuousDistribution_T&,
00184                 const DemandDistribution&,
00185                 stdair::BaseGenerator_T& ioSharedGenerator,
00186                 const stdair::RandomSeed_T& iRequestDateTimeSeed,
00187                 const stdair::RandomSeed_T& iDemandCharacteristicsSeed,
00188                 const POSProbabilityMass_T&);
00189
00194     void setBoolFirstDateTimeRequest (const bool& iFirstDateTimeRequest) {
00195         _firstDateTimeRequest = iFirstDateTimeRequest;
00196     }
00197
00198 public:
00199     // ////////////////////////////////// Business Methods //////////////////////////////////
00202     void incrementGeneratedRequestsCounter() {
00203         _randomGenerationContext.incrementGeneratedRequestsCounter();
00204     }
00205
00207     const bool stillHavingRequestsToBeGenerated (const
stdair::DemandGenerationMethod& iDemandGenerationMethod) const;
00208
00210     const stdair::DateTime_T generateTimeOfRequestPoissonProcess();
00211
00213     const stdair::DateTime_T generateTimeOfRequestStatisticsOrder();
00214
00216     const stdair::AirportCode_T generatePOS();
00217
00219     const stdair::ChannelLabel_T generateChannel();
00220
00222     const stdair::TripType_T generateTripType();
00223
00225     const stdair::DayDuration_T generateStayDuration();
00226
00228     const stdair::FrequentFlyer_T generateFrequentFlyer();
00229
00231     const stdair::Duration_T generatePreferredDepartureTime();
00232
00234     const stdair::WTP_T generateWTP (stdair::RandomGeneration&,
00235                                     const stdair::Date_T&,
00236                                     const stdair::DateTime_T&,
00237                                     const stdair::DayDuration_T&);
00238
00240     const stdair::PriceValue_T generateValueOfTime();
00241
00252     stdair::BookingRequestPtr_T
00253     generateNextRequest (stdair::RandomGeneration&,
00254                         const stdair::DemandGenerationMethod&);
00255
00257     void reset (stdair::BaseGenerator_T& ioSharedGenerator);
00258
00259 public:
00260     // ////////////////////////////////// Display support methods //////////////////////////////////
00266     void toStream (std::ostream& ioOut) const {
00267         ioOut << toString();
00268     }
00269
00274     void fromStream (std::istream& ioIn) {
00275     }
00276
00280     std::string toString() const;
00281
00285     const std::string describeKey() const {
00286         return _key.toString();
00287     }
00288
00292     std::string display() const;
00293
00294     const stdair::Duration_T convertFloatIntoDuration (const
stdair::FloatDuration_T);
00295
00296 protected:
00297     // ////////////////////////////////// Constructors and destructors //////////////////////////////////
00301     DemandStream (const Key_T&);
00305     virtual ~DemandStream();
00306
00307 private:

```

```

00309     DemandStream();
00311     DemandStream (const DemandStream&);
00313     void init (stdair::BaseGenerator_T& ioSharedGenerator);
00314
00315
00316 protected:
00317     // ////////// Attributes //////////
00321     Key_T _key;
00322
00326     BomAbstract* _parent;
00327
00331     stdair::HolderMap_T _holderMap;
00332
00336     DemandCharacteristics _demandCharacteristics;
00337
00341     DemandDistribution _demandDistribution;
00342
00346     stdair::NbOfRequests_T _totalNumberOfRequestsToBeGenerated;
00347
00351     RandomGenerationContext _randomGenerationContext;
00352
00356     stdair::RandomGeneration _requestDateTimeRandomGenerator;
00357
00361     stdair::RandomGeneration _demandCharacteristicsRandomGenerator;
00362
00367     POSProbabilityMass_T _posProMass;
00368
00369 private:
00370     bool _stillHavingRequestsToBeGenerated;
00371
00372     bool _firstDateTimeRequest;
00373
00374     stdair::FloatDuration_T _dateTimeLastRequest;
00375 };
00376
00377 }
00378 #endif // __TRADEMGEN_BOM_DEMANDSTREAM_HPP

```

25.69 trademgen/bom/DemandStreamKey.cpp File Reference

```

#include <cassert>
#include <sstream>
#include <stdair/basic/BasConst_Inventory.hpp>
#include <trademgen/bom/DemandStreamKey.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.70 DemandStreamKey.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // StdAir
00008 #include <stdair/basic/BasConst_Inventory.hpp>
00009 // TraDemGen
00010 #include <trademgen/bom/DemandStreamKey.hpp>
00011
00012 namespace TRADEMGEN {
00013
00014     // //////////////////////////////////////
00015     DemandStreamKey::DemandStreamKey ()
00016         : _origin (stdair::DEFAULT_ORIGIN),
00017           _destination (stdair::DEFAULT_DESTINATION),
00018           _preferredDepartureDate (stdair::DEFAULT_DEPARTURE_DATE),
00019           _preferredCabin (stdair::DEFAULT_CABIN_CODE) {
00020         assert (false);
00021     }
00022
00023     // //////////////////////////////////////
00024     DemandStreamKey::

```

```

00025 DemandStreamKey (const stdair::AirportCode_T& iOrigin,
00026                  const stdair::AirportCode_T& iDestination,
00027                  const stdair::Date_T& iPreferredDepartureDate,
00028                  const stdair::CabinCode_T& iPreferredCabin)
00029 : _origin (iOrigin), _destination (iDestination),
00030   _preferredDepartureDate (iPreferredDepartureDate),
00031   _preferredCabin (iPreferredCabin) {
00032 }
00033
00034 // //////////////////////////////////////
00035 DemandStreamKey::DemandStreamKey (const DemandStreamKey& iKey)
00036 : _origin (iKey._origin), _destination (iKey._destination),
00037   _preferredDepartureDate (iKey._preferredDepartureDate),
00038   _preferredCabin (iKey._preferredCabin) {
00039 }
00040
00041 // //////////////////////////////////////
00042 DemandStreamKey::~DemandStreamKey () {
00043 }
00044
00045 // //////////////////////////////////////
00046 void DemandStreamKey::toStream (std::ostream& ioOut) const {
00047     ioOut << "DemandStreamKey: " << toString();
00048 }
00049
00050 // //////////////////////////////////////
00051 void DemandStreamKey::fromStream (std::istream& ioIn) {
00052 }
00053
00054 // //////////////////////////////////////
00055 const std::string DemandStreamKey::toString() const {
00056     std::ostringstream oStr;
00057     oStr << _origin << "-" << _destination << " " << _preferredDepartureDate
00058         << " " << _preferredCabin;
00059     return oStr.str();
00060 }
00061
00062 }

```

25.71 trademgen/bom/DemandStreamKey.hpp File Reference

```

#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_date_time_types.hpp>
#include <stdair/bom/KeyAbstract.hpp>

```

Classes

- struct [TRADEMGEN::DemandStreamKey](#)

Namespaces

- namespace [TRADEMGEN](#)

25.72 DemandStreamKey.hpp

```

00001 #ifndef __TRADEMGEN_BOM_DEMANDSTREAMKEY_HPP
00002 #define __TRADEMGEN_BOM_DEMANDSTREAMKEY_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/stdair_date_time_types.hpp>
00010 #include <stdair/bom/KeyAbstract.hpp>
00011
00012 namespace TRADEMGEN {
00013
00020     struct DemandStreamKey : public stdair::KeyAbstract {
00021
00022         // ////////////////////////////////// Constructors and destructors //////////////////////////////////
00023     private:

```

```

00025     DemandStreamKey();
00026
00027 public:
00029     DemandStreamKey (const stdair::AirportCode_T& iOrigin,
00030                     const stdair::AirportCode_T& iDestination,
00031                     const stdair::Date_T& iPreferredDepartureDate,
00032                     const stdair::CabinCode_T& iPreferredCabin);
00034     DemandStreamKey (const DemandStreamKey&);
00035
00037     ~DemandStreamKey();
00038
00039
00040 public:
00041     // //////////// Getters ////////////
00043     const stdair::AirportCode_T& getOrigin() const {
00044         return _origin;
00045     }
00046
00048     const stdair::AirportCode_T& getDestination() const {
00049         return _destination;
00050     }
00051
00053     const stdair::Date_T& getPreferredDepartureDate () const {
00054         return _preferredDepartureDate;
00055     }
00056
00058     const stdair::CabinCode_T& getPreferredCabin() const {
00059         return _preferredCabin;
00060     }
00061
00062
00063     // //////////// Display support methods ////////////
00066     void toStream (std::ostream& ioOut) const;
00067
00070     void fromStream (std::istream& ioIn);
00071
00077     const std::string toString() const;
00078
00079 private:
00081     // //////////// Attributes ////////////
00083     stdair::AirportCode_T _origin;
00084
00086     stdair::AirportCode_T _destination;
00087
00089     stdair::Date_T _preferredDepartureDate;
00090
00092     stdair::CabinCode_T _preferredCabin;
00093 };
00094
00095 }
00096 #endif // __TRADEMGEN_BOM_DEMANDSTREAMKEY_HPP

```

25.73 trademgen/bom/DemandStreamTypes.hpp File Reference

```

#include <map>
#include <list>
#include <stdair/bom/key_types.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef std::list< DemandStream * > [TRADEMGEN::DemandStreamList_T](#)
- typedef std::map< const
stdair::MapKey_T, DemandStream * > [TRADEMGEN::DemandStreamMap_T](#)

25.74 DemandStreamTypes.hpp

```

00001 #ifndef __TRADEMGEN_BOM_DEMANDSTREAMTYPES_HPP

```

```

00002 #define __TRADEMGEN_BOM_DEMANDSTREAMTYPES_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <map>
00009 #include <list>
00010 // StdAir
00011 #include <stdair/bom/key_types.hpp>
00012
00013 namespace TRADEMGEN {
00014
00015     // Forward declarations.
00016     class DemandStream;
00017
00018     typedef std::list<DemandStream*> DemandStreamList_T;
00019
00020
00022     typedef std::map<const stdair::MapKey_T, DemandStream*> DemandStreamMap_T;
00023
00024 }
00025 #endif // __TRADEMGEN_BOM_DEMANDSTREAMTYPES_HPP

```

25.75 trademgen/bom/DemandStruct.cpp File Reference

```

#include <cassert>
#include <sstream>
#include <stdair/basic/BasConst_Inventory.hpp>
#include <stdair/basic/BasConst_Period_BOM.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
#include <trademgen/bom/DemandStruct.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.76 DemandStruct.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // StdAir
00008 #include <stdair/basic/BasConst_Inventory.hpp>
00009 #include <stdair/basic/BasConst_Period_BOM.hpp>
00010 #include <stdair/service/Logger.hpp>
00011 // TRADEMGEN
00012 #include <trademgen/TRADEMGEN_Types.hpp>
00013 #include <trademgen/bom/DemandStruct.hpp>
00014
00015 namespace TRADEMGEN {
00016
00017     // //////////////////////////////////////
00018     DemandStruct::DemandStruct ()
00019     : _dateRange (stdair::BOOST_DEFAULT_DATE_PERIOD),
00020       _dow (stdair::DEFAULT_DOW_STRING),
00021       _prefCabin (stdair::DEFAULT_CABIN_CODE),
00022       _itHours (0), _itMinutes (0), _itSeconds (0), _itFFCode ("") {
00023     }
00024
00025     // //////////////////////////////////////
00026     DemandStruct::~DemandStruct () {
00027     }
00028
00029     // //////////////////////////////////////
00030     stdair::Date_T DemandStruct::getDate() const {
00031         return stdair::Date_T (_itYear, _itMonth, _itDay);
00032     }
00033
00034     // //////////////////////////////////////

```

```

00035 stdair::Duration_T DemandStruct::getTime() const {
00036     return boost::posix_time::hours (_itHours)
00037         + boost::posix_time::minutes (_itMinutes)
00038         + boost::posix_time::seconds (_itSeconds);
00039 }
00040
00041 // //////////////////////////////////////
00042 const std::string DemandStruct::describe() const {
00043     std::ostringstream ostr;
00044     ostr << _dateRange << " - " << _dow
00045         << " " << _origin << " - " << _destination
00046         << " " << _prefCabin
00047         << ", N(" << _demandMean << ", " << _demandStdDev << "); ";
00048
00049     unsigned short idx = 0;
00050     for (POSProbabilityMassFunction_T::const_iterator it = _posProbDist.begin()
;
00051         it != _posProbDist.end(); ++it, ++idx) {
00052         const stdair::AirportCode_T& lPosCode = it->first;
00053         const stdair::Probability_T& lPosProbMass = it->second;
00054         if (idx != 0) {
00055             ostr << ", ";
00056         }
00057         ostr << lPosCode << ":" << lPosProbMass;
00058     }
00059     ostr << "; ";
00060
00061     idx = 0;
00062     for (ChannelProbabilityMassFunction_T::const_iterator it =
         _channelProbDist.begin();
00063         it != _channelProbDist.end(); ++it, ++idx) {
00064         const stdair::ChannelLabel_T lChannelCode = it->first;
00065         const stdair::Probability_T& lChannelProbMass = it->second;
00066         if (idx != 0) {
00067             ostr << ", ";
00068         }
00069         ostr << lChannelCode << ":" << lChannelProbMass;
00070     }
00071     ostr << "; ";
00072
00073     idx = 0;
00074     for (TripTypeProbabilityMassFunction_T::const_iterator it =
         _tripProbDist.begin();
00075         it != _tripProbDist.end(); ++it, ++idx) {
00076         const stdair::TripType_T lTripCode = it->first;
00077         const stdair::Probability_T& lTripProbMass = it->second;
00078         if (idx != 0) {
00079             ostr << ", ";
00080         }
00081         ostr << lTripCode << ":" << lTripProbMass;
00082     }
00083     ostr << "; ";
00084
00085     idx = 0;
00086     for (StayDurationProbabilityMassFunction_T::const_iterator it =
         _stayProbDist.begin();
00087         it != _stayProbDist.end(); ++it, ++idx) {
00088         const stdair::DayDuration_T& lStayDuration = it->first;
00089         const stdair::Probability_T& lStayProbMass = it->second;
00090         if (idx != 0) {
00091             ostr << ", ";
00092         }
00093         ostr << lStayDuration << ":" << lStayProbMass;
00094     }
00095     ostr << "; ";
00096
00097     idx = 0;
00098     for (FrequentFlyerProbabilityMassFunction_T::const_iterator it =
         _ffProbDist.begin();
00099         it != _ffProbDist.end(); ++it, ++idx) {
00100         const stdair::FrequentFlyer_T lFFCode = it->first;
00101         const stdair::Probability_T& lFFProbMass = it->second;
00102         if (idx != 0) {
00103             ostr << ", ";
00104         }
00105         ostr << lFFCode << ":" << lFFProbMass;
00106     }
00107     ostr << "; ";
00108
00109     idx = 0;
00110     for (PreferredDepartureTimeContinuousDistribution_T::const_iterator it =
         _prefDepTimeProbDist.begin();
00111         it != _prefDepTimeProbDist.end(); ++it, ++idx) {
00112         const stdair::IntDuration_T& lPrefDepTime = it->first;
00113         const stdair::Probability_T& lPrefDepTimeProbMass = it->second;
00114         if (idx != 0) {
00115             ostr << ", ";
00116         }
00117         ostr << lPrefDepTime << ":" << lPrefDepTimeProbMass;
00118     }
00119     ostr << "; ";
00120

```

```

00121     }
00122     ostr << lPrefDepTime << ":" << lPrefDepTimeProbMass;
00123 }
00124 ostr << "; ";
00125
00126 ostr << _minWTP << "; ";
00127
00128 idx = 0;
00129 for (ValueOfTimeContinuousDistribution_T::const_iterator it =
00130     _timeValueProbDist.begin();
00131     it != _timeValueProbDist.end(); ++it, ++idx) {
00132     const stdair::PriceValue_T& lTimeValue = it->first;
00133     const stdair::Probability_T& lTimeValueProbMass = it->second;
00134     if (idx != 0) {
00135         ostr << ", ";
00136     }
00137     ostr << lTimeValue << ":" << lTimeValueProbMass;
00138 }
00139 ostr << "; ";
00140
00141 idx = 0;
00142 for (ArrivalPatternCumulativeDistribution_T::const_iterator it =
00143     _dtdProbDist.begin(); it != _dtdProbDist.end(); ++it, ++idx) {
00144     const stdair::FloatDuration_T& lDTD = it->first;
00145     const stdair::Probability_T& lDTDProbMass = it->second;
00146     if (idx != 0) {
00147         ostr << ", ";
00148     }
00149     ostr << lDTD << ":" << lDTDProbMass;
00150 }
00151 ostr << "; ";
00152
00153 return ostr.str();
00154 }
00155
00156 }

```

25.77 trademgen/bom/DemandStruct.hpp File Reference

```

#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_maths_types.hpp>
#include <stdair/stdair_date_time_types.hpp>
#include <stdair/basic/StructAbstract.hpp>
#include <stdair/bom/DoWStruct.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>

```

Classes

- struct [TRADEMGEN::DemandStruct](#)

Namespaces

- namespace [TRADEMGEN](#)

25.78 DemandStruct.hpp

```

00001 #ifndef __TRADEMGEN_BOM_DEMANDSTRUCT_HPP
00002 #define __TRADEMGEN_BOM_DEMANDSTRUCT_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_basic_types.hpp>
00011 #include <stdair/stdair_maths_types.hpp>
00012 #include <stdair/stdair_date_time_types.hpp>
00013 #include <stdair/basic/StructAbstract.hpp>

```

```

00014 #include <stdair/bom/DoWStruct.hpp>
00015 // TraDemGen
00016 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00017
00018 namespace TRADEMGEN {
00019
00021     struct DemandStruct : public stdair::StructAbstract {
00022
00023     public:
00024         // ////////////////////////////////// Getters //////////////////////////////////
00026         stdair::Date_T getDate() const;
00027
00029         stdair::Duration_T getTime() const;
00030
00031
00032     public:
00033         // ////////////////////////////////// Display Support Methods //////////////////////////////////
00035         const std::string describe() const;
00036
00037
00038     public:
00039         // ////////////////////////////////// Constructors and destructors //////////////////////////////////
00041         DemandStruct();
00043         ~DemandStruct();
00044     private:
00046         DemandStruct (const DemandStruct&);
00047
00048
00049     public:
00050         // ////////////////////////////////// Attributes //////////////////////////////////
00051         stdair::DatePeriod_T _dateRange;
00052         stdair::DoWStruct _dow;
00053         stdair::AirportCode_T _origin;
00054         stdair::AirportCode_T _destination;
00055         stdair::CabinCode_T _prefCabin;
00056         stdair::MeanValue_T _demandMean;
00057         stdair::StdDevValue_T _demandStdDev;
00058         POSProbabilityMassFunction_T _posProbDist;
00059         ChannelProbabilityMassFunction_T _channelProbDist;
00060         TripTypeProbabilityMassFunction_T _tripProbDist;
00061         StayDurationProbabilityMassFunction_T _stayProbDist;
00062         FrequentFlyerProbabilityMassFunction_T _ffProbDist;
00063         PreferredDepartureTimeContinuousDistribution_T _prefDepTimeProbDist;
00064         stdair::WTP_T _minWTP;
00065         ValueOfTimeContinuousDistribution_T _timeValueProbDist;
00066         ArrivalPatternCumulativeDistribution_T _dtdProbDist;
00067
00068     public:
00069         // ////////////////////////////////// Staging //////////////////////////////////
00071         stdair::Date_T _prefDepDateStart;
00072         stdair::Date_T _prefDepDateEnd;
00073         unsigned int _itYear;
00074         unsigned int _itMonth;
00075         unsigned int _itDay;
00076
00078         long _itHours;
00079         long _itMinutes;
00080         long _itSeconds;
00081
00083         stdair::AirportCode_T _itPosCode;
00084
00086         stdair::ChannelLabel_T _itChannelCode;
00087
00089         stdair::TripType_T _itTripCode;
00090
00092         stdair::DayDuration_T _itStayDuration;
00093
00095         stdair::FrequentFlyer_T _itFFCode;
00096
00098         stdair::Duration_T _itPrefDepTime;
00099
00101         stdair::PriceValue_T _itTimeValue;
00102
00104         stdair::DayDuration_T _itDTD;
00105     };
00106
00107 }
00108 #endif // __TRADEMGEN_BOM_DEMANDSTRUCT_HPP

```

25.79 trademgen/command/DBManager.cpp File Reference

```
#include <cassert>
```



```
#include <soci/soci.h>
#include <soci/mysql/soci-mysql.h>
#include <stdair/bom/AirlineStruct.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/command/DBManager.hpp>
```

Namespaces

- namespace **TRADEMGEN**

25.80 DBManager.cpp

```
00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // SOCI
00007 #if defined(SOCI_HEADERS_BURIED)
00008 #include <soci/core/soci.h>
00009 #include <soci/backends/mysql/soci-mysql.h>
00010 #else // SOCI_HEADERS_BURIED
00011 #include <soci/soci.h>
00012 #include <soci/mysql/soci-mysql.h>
00013 #endif // SOCI_HEADERS_BURIED
00014 // StdAir
00015 #include <stdair/bom/AirlineStruct.hpp>
00016 #include <stdair/service/Logger.hpp>
00017 // TraDemGen
00018 #include <trademgen/command/DBManager.hpp>
00019
00020 namespace TRADEMGEN {
00021
00022 // //////////////////////////////////////
00023 void DBManager::
00024     prepareSelectStatement (stdair::DBSession_T& ioSociSession,
00025                             stdair::DBRequestStatement_T& ioSelectStatement,
00026                             stdair::AirlineStruct& ioAirline) {
00027
00028     try {
00029
00030         // Instantiate a SQL statement (no request is performed at that stage)
00044     } catch (std::exception const& lException) {
00045         STDAIR_LOG_ERROR ("Error: " << lException.what());
00046         throw stdair::SQLDatabaseException (lException.what());
00047     }
00048 }
00049
00050 // //////////////////////////////////////
00051 void DBManager::
00052     prepareSelectOnAirlineCodeStatement (stdair::DBSession_T& ioSociSession,
00053                                           stdair::DBRequestStatement_T&
00054                                           ioSelectStatement,
00055                                           const stdair::AirlineCode_T&
00056                                           iAirlineCode,
00057                                           stdair::AirlineStruct& ioAirline) {
00058
00059     try {
00060
00061         // Instantiate a SQL statement (no request is performed at that stage)
00090     } catch (std::exception const& lException) {
00091         STDAIR_LOG_ERROR ("Error: " << lException.what());
00092         throw stdair::SQLDatabaseException (lException.what());
00093     }
00094 }
00095
00096 // //////////////////////////////////////
00097 bool DBManager::iterateOnStatement (stdair::DBRequestStatement_T& ioStatement
00098                                     ,
00099                                     stdair::AirlineStruct& ioAirline,
00100                                     const bool iShouldDoReset) {
00101
00102     bool hasStillData = false;
00103
00104     try {
00105
00106         // Reset the list of names of the given Airline object
00107         if (iShouldDoReset == true) {
00108             // ioAirline.resetMatrix();
00109         }
00110     }
00111 }
```

```

00107     }
00108
00109     // Retrieve the next row of Airline object
00110     hasStillData = ioStatement.fetch();
00111
00112     } catch (std::exception const& lException) {
00113         STDAIR_LOG_ERROR ("Error: " << lException.what());
00114         throw stdair::SQLDatabaseException (lException.what());
00115     }
00116
00117     return hasStillData;
00118 }
00119
00120 //////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
00121 void DBManager::updateAirlineInDB (stdair::DBSession_T& ioSociSession,
00122                                   const stdair::AirlineStruct& iAirline) {
00123
00124     try {
00125
00126         // Begin a transaction on the database
00127         ioSociSession.begin();
00128
00129         // Instanciate a SQL statement (no request is performed at that stage)
00130         std::string lAirlineCode;
00131         /*
00132         stdair::DBRequestStatement_T lUpdateStatement =
00133             (ioSociSession.prepare
00134              << "update ref_airline_details "
00135              << "set xapien_docid = :xapien_docid "
00136              << "where code = :code", soci::use (lDocID), soci::use
(lAirlineCode));
00137
00138         // Execute the SQL query
00139         lDocID = iAirline.getDocID();
00140         lAirlineCode = iAirline.getAirlineCode();
00141         lUpdateStatement.execute (true);
00142         */
00143
00144         // Commit the transaction on the database
00145         ioSociSession.commit();
00146
00147         // Debug
00148         // TRADEMGEN_LOG_DEBUG ("[" << lDocID << "]" << iAirline);
00149
00150     } catch (std::exception const& lException) {
00151         STDAIR_LOG_ERROR ("Error: " << lException.what());
00152         throw stdair::SQLDatabaseException (lException.what());
00153     }
00154 }
00155
00156 //////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
00157 bool DBManager::retrieveAirline (stdair::DBSession_T& ioSociSession,
00158                                 const stdair::AirlineCode_T& iAirlineCode,
00159                                 stdair::AirlineStruct& ioAirline) {
00160     bool oHasRetrievedAirline = false;
00161
00162     try {
00163
00164         // Prepare the SQL request corresponding to the select statement
00165         stdair::DBRequestStatement_T lSelectStatement (ioSociSession);
00166         DBManager::prepareSelectOnAirlineCodeStatement (ioSociSession,
00167                                                         lSelectStatement,
00168                                                         iAirlineCode, ioAirline);
00169
00170         const bool shouldDoReset = true;
00171         bool hasStillData = iterateOnStatement (lSelectStatement, ioAirline,
00172                                                shouldDoReset);
00173
00174         if (hasStillData == true) {
00175             oHasRetrievedAirline = true;
00176         }
00177
00178         // Sanity check
00179         const bool shouldNotDoReset = false;
00180         hasStillData = iterateOnStatement (lSelectStatement, ioAirline,
00181                                          shouldNotDoReset);
00182
00183         // Debug
00184         // STDAIR_LOG_DEBUG ("[" << iDocID << "]" << ioAirline);
00185
00186     } catch (std::exception const& lException) {
00187         STDAIR_LOG_ERROR ("Error: " << lException.what());
00188         throw stdair::SQLDatabaseException (lException.what());
00189     }
00190
00191     return oHasRetrievedAirline;
00192 }
00193 }

```

25.81 trademgen/command/DBManager.hpp File Reference

```
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_db.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
```

Classes

- class [TRADEMGEN::DBManager](#)

Namespaces

- namespace [TRADEMGEN](#)

25.82 DBManager.hpp

```
00001 #ifndef __TRADEMGEN_CMD_DBMANAGER_HPP
00002 #define __TRADEMGEN_CMD_DBMANAGER_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/stdair_db.hpp>
00010 // Trademgen
00011 #include <trademgen/TRADEMGEN_Types.hpp>
00012
00013 namespace TRADEMGEN {
00014
00015     // Forward declarations
00016     struct AirlineStruct;
00017
00020     class DBManager {
00021     public:
00024         static void updateAirlineInDB (stdair::DBSession_T&,
00025                                         const stdair::AirlineStruct&);
00026
00030         static bool retrieveAirline (stdair::DBSession_T&,
00031                                     const stdair::AirlineCode_T&,
00032                                     stdair::AirlineStruct&);
00033
00034
00035     public:
00037         static void prepareSelectStatement (stdair::DBSession_T&,
00038                                             stdair::DBRequestStatement_T&,
00039                                             stdair::AirlineStruct&);
00040
00045         static bool iterateOnStatement (stdair::DBRequestStatement_T&,
00046                                         stdair::AirlineStruct&,
00047                                         const bool iShouldDoReset);
00048
00049
00050     private:
00052         static void prepareSelectOnAirlineCodeStatement (stdair::DBSession_T&,
00053                                                         stdair::DBRequestStatement_T&,
00054                                                         const
00055                                                         stdair::AirlineCode_T&,
00056                                                         stdair::AirlineStruct&);
00057
00058     private:
00060         DBManager() {}
00061         DBManager(const DBManager&) {}
00063         ~DBManager() {}
00064     };
00065
00066 }
00067 #endif // __TRADEMGEN_CMD_DBMANAGER_HPP
```

25.83 trademgen/command/DemandManager.cpp File Reference

```

#include <cassert>
#include <stdair/basic/ProgressStatusSet.hpp>
#include <stdair/bom/BomManager.hpp>
#include <stdair/bom/EventStruct.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/bom/EventQueue.hpp>
#include <stdair/bom/TravelSolutionStruct.hpp>
#include <stdair/bom/CancellationStruct.hpp>
#include <stdair/factory/FacBom.hpp>
#include <stdair/factory/FacBomManager.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/basic/DemandCharacteristics.hpp>
#include <trademgen/basic/DemandDistribution.hpp>
#include <trademgen/bom/DemandStruct.hpp>
#include <trademgen/bom/DemandStream.hpp>
#include <trademgen/command/DemandManager.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.84 DemandManager.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // StdAir
00007 #include <stdair/basic/ProgressStatusSet.hpp>
00008 #include <stdair/bom/BomManager.hpp>
00009 #include <stdair/bom/EventStruct.hpp>
00010 #include <stdair/bom/BookingRequestStruct.hpp>
00011 #include <stdair/bom/EventQueue.hpp>
00012 #include <stdair/bom/TravelSolutionStruct.hpp>
00013 #include <stdair/bom/CancellationStruct.hpp>
00014 #include <stdair/factory/FacBom.hpp>
00015 #include <stdair/factory/FacBomManager.hpp>
00016 #include <stdair/service/Logger.hpp>
00017 // TraDemGen
00018 #include <trademgen/basic/DemandCharacteristics.hpp>
00019 #include <trademgen/basic/DemandDistribution.hpp>
00020 #include <trademgen/bom/DemandStruct.hpp>
00021 #include <trademgen/bom/DemandStream.hpp>
00022 #include <trademgen/command/DemandManager.hpp>
00023
00024 namespace TRADEMGEN {
00025
00026 // //////////////////////////////////////
00027 void DemandManager::
00028     buildSampleBomStd (stdair::EventQueue& ioEventQueue,
00029                       stdair::RandomGeneration& ioSharedGenerator,
00030                       const POSProbabilityMass_T& iPOSProbMass) {
00031
00032     // Key of the demand stream
00033     const stdair::AirportCode_T lOrigin ("SIN");
00034     const stdair::AirportCode_T lDestination ("BKK");
00035     const stdair::Date_T lDepDate (2011, 2, 14);
00036     const stdair::CabinCode_T lCabin ("Y");
00037
00038     //
00039     const DemandStreamKey lDemandStreamKey (lOrigin, lDestination, lDepDate,
00040                                             lCabin);
00041
00042     // DEBUG
00043     // STDAIR_LOG_DEBUG ("Demand stream key: " << lDemandStreamKey.describe());
00044
00045     // Distribution for the number of requests
00046     const stdair::MeanValue_T lDemandMean (10.0);

```

```

00047     const stdair::StdDevValue_T lDemandStdDev (1.0);
00048     const DemandDistribution lDemandDistribution (lDemandMean, lDemandStdDev);
00049
00050     // Seed
00051     const stdair::RandomSeed_T& lRequestDateTimeSeed =
00052         generateSeed (ioSharedGenerator);
00053     const stdair::RandomSeed_T& lDemandCharacteristicsSeed =
00054         generateSeed (ioSharedGenerator);
00055
00056     //
00057     ArrivalPatternCumulativeDistribution_T lDTProbDist;
00058     lDTProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-330
00059
00059         0));
00060     lDTProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-40,
00061         0.2));
00062 );
00063     lDTProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-20,
00064         0.6));
00065 );
00066     lDTProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-1,
00067         1.0));
00068 );
00069     //
00070     POSProbabilityMassFunction_T lPOSProbDist;
00071     lPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("BKK", 0.3))
00072 ;
00073     lPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("SIN", 0.7))
00074 ;
00075     //
00076     ChannelProbabilityMassFunction_T lChannelProbDist;
00077     lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("DF"
00078         0.1));
00079 );
00080     lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("DN"
00081         0.3));
00082 );
00083     lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("IF"
00084         0.4));
00085 );
00086     lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("IN"
00087         0.2));
00088 );
00089     //
00090     TripTypeProbabilityMassFunction_T lTripProbDist;
00091     lTripProbDist.insert (TripTypeProbabilityMassFunction_T::value_type ("RO",
00092         0.6));
00093     lTripProbDist.insert (TripTypeProbabilityMassFunction_T::value_type ("RI",
00094         0.2));
00095     lTripProbDist.insert (TripTypeProbabilityMassFunction_T::value_type ("OW",
00096         0.2));
00097 //
00098     StayDurationProbabilityMassFunction_T lStayProbDist;
00099     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (0,
00100         0.1));
00101 );
00102     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (1,
00103         0.1));
00104 );
00105     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (2,
00106         .15));
00107 );
00108     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (3,
00109         .15));
00110 );
00111     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (4,
00112         .15));
00113 );
00114     lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (5,
00115         .35));
00116 );
00117     //
00118     FrequentFlyerProbabilityMassFunction_T lFFProbDist;
00119     lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type ("P",
00120         0.01));
00121 );
00122     lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type ("G",
00123         0.05));
00124 );
00125     lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type ("S",
00126         0.15));
00127 );
00128     lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type ("M",

```

```

00111                                                                 0.3))
00112 ;
00113     lFFProbDist.insert(FrequentFlyerProbabilityMassFunction_T::value_type("N",
00114                                                                 0.49)
00115 );
00116 //
00117 PreferredDepartureTimeContinuousDistribution_T lPrefDepTimeProbDist;
00118 lPrefDepTimeProbDist.
00119     insert (PreferredDepartureTimeContinuousDistribution_T::value_type (6, 0)
00120 );
00121 lPrefDepTimeProbDist.
00122     insert (PreferredDepartureTimeContinuousDistribution_T::value_type (7,
00123                                                                 0.1))
00124 ;
00125 lPrefDepTimeProbDist.
00126     insert (PreferredDepartureTimeContinuousDistribution_T::value_type (9,
00127                                                                 0.3))
00128 ;
00129 lPrefDepTimeProbDist.
00130     insert (PreferredDepartureTimeContinuousDistribution_T::value_type (17,
00131                                                                 0.4))
00132 ;
00133 lPrefDepTimeProbDist.
00134     insert (PreferredDepartureTimeContinuousDistribution_T::value_type (19,
00135                                                                 0.80)
00136 );
00137 lPrefDepTimeProbDist.
00138     insert (PreferredDepartureTimeContinuousDistribution_T::value_type (20,
00139                                                                 0.95)
00140 );
00141 lPrefDepTimeProbDist.
00142     insert (PreferredDepartureTimeContinuousDistribution_T::value_type (22,
00143                                                                 1));
00144 //
00145 ValueOfTimeContinuousDistribution_T lTimeValueProbDist;
00146 lTimeValueProbDist.insert(ValueOfTimeContinuousDistribution_T::value_type(1
00147 5,
00148                                                                 0
00149 ));
00150 lTimeValueProbDist.insert(ValueOfTimeContinuousDistribution_T::value_type(6
00151 0,
00152                                                                 1
00153 ));
00154 //
00155 const stdair::WTP_T lWTP (1000.0);
00156
00157 // Delegate the call to the dedicated command
00158 DemandStream& lDemandStream =
00159     createDemandStream (ioEventQueue, lDemandStreamKey, lDTPProbDist,
00160                         lPOSProbDist, lChannelProbDist, lTripProbDist,
00161                         lStayProbDist, lFFProbDist, lPrefDepTimeProbDist,
00162                         lWTP, lTimeValueProbDist, lDemandDistribution,
00163                         ioSharedGenerator.getBaseGenerator(),
00164                         lRequestDateTimeSeed,
00165                         lDemandCharacteristicsSeed, iPOSProbMass);
00166
00167 // Calculate the expected total number of events for the current
00168 // demand stream
00169 const stdair::NbOfRequests_T& lExpectedTotalNbOfEvents =
00170     lDemandStream.getMeanNumberOfRequests();
00171
00172 ioEventQueue.addStatus (stdair::EventType::BKG_REQ,
00173                         lExpectedTotalNbOfEvents);
00174 }
00175 // //////////////////////////////////////
00176 DemandStream& DemandManager::createDemandStream
00177 (stdair::EventQueue& ioEventQueue,
00178  const DemandStreamKey& iKey,
00179  const ArrivalPatternCumulativeDistribution_T& iArrivalPattern,
00180  const POSProbabilityMassFunction_T& iPOSProbMass,
00181  const ChannelProbabilityMassFunction_T& iChannelProbMass,
00182  const TripTypeProbabilityMassFunction_T& iTripTypeProbMass,
00183  const StayDurationProbabilityMassFunction_T& iStayDurationProbMass,
00184  const FrequentFlyerProbabilityMassFunction_T& iFrequentFlyerProbMass,
00185  const PreferredDepartureTimeContinuousDistribution_T&
00186  iPreferredDepartureTimeContinuousDistribution,
00187  const stdair::WTP_T& iMinWTP,
00188  const ValueOfTimeContinuousDistribution_T&
00189  iValueOfTimeContinuousDistribution,
00190  const DemandDistribution& iDemandDistribution,
00191  stdair::BaseGenerator_T& ioSharedGenerator,
00192  const stdair::RandomSeed_T& iRequestDateTimeSeed,
00193  const stdair::RandomSeed_T& iDemandCharacteristicsSeed,
00194  const POSProbabilityMass_T& iDefaultPOSProbabilityMass) {

```

```

00187
00188     DemandStream& oDemandStream =
00189         stdair::FacBom<DemandStream>::instance().create (iKey);
00190
00191     oDemandStream.setAll (iArrivalPattern, iPOSProbMass,
00192         iChannelProbMass, iTripTypeProbMass,
00193         iStayDurationProbMass, iFrequentFlyerProbMass,
00194         iPreferredDepartureTimeContinuousDistribution,
00195         iMinWTP, iValueOfTimeContinuousDistribution,
00196         iDemandDistribution, ioSharedGenerator,
00197         iRequestDateTimeSeed, iDemandCharacteristicsSeed,
00198         iDefaultPOSProbabilityMass);
00199
00200     // Link the DemandStream to its parent (EventQueue)
00201     stdair::FacBomManager::linkWithParent (ioEventQueue, oDemandStream);
00202
00203     // Add the DemandStream to the dedicated list and map
00204     stdair::FacBomManager::addToListAndMap (ioEventQueue, oDemandStream);
00205
00206     return oDemandStream;
00207 }
00208
00209 // //////////////////////////////////////
00210 void DemandManager::
00211 createDemandCharacteristics (stdair::EventQueue& ioEventQueue,
00212     stdair::RandomGeneration& ioSharedGenerator,
00213     const POSProbabilityMass_T& iPOSProbMass,
00214     const DemandStruct& iDemand) {
00215     stdair::BaseGenerator_T& lSharedGenerator =
00216         ioSharedGenerator.getBaseGenerator();
00217
00218     // Parse the date period and DoW and generate demand characteristics.
00219     const stdair::DatePeriod_T lDateRange = iDemand._dateRange;
00220     for (boost::gregorian::day_iterator itDate = lDateRange.begin();
00221         itDate != lDateRange.end(); ++itDate) {
00222         const stdair::Date_T& currentDate = *itDate;
00223
00224         // Retrieve, for the current day, the Day-Of-the-Week (thanks to Boost)
00225         const unsigned short currentDoW = currentDate.day_of_week().as_number();
00226
00227         // The demand structure stores which Days (-Of-the-Week) are
00228         // active within the week. For each day (Mon., Tue., etc.), a boolean
00229         // states whether the Flight is active for that day.
00230         const stdair::DoWStruct& lDoWList = iDemand._dow;
00231         const bool isDoWActive = lDoWList.getStandardDayOfWeek (currentDoW);
00232
00233         if (isDoWActive == true) {
00234             const DemandStreamKey lDemandStreamKey (iDemand._origin,
00235                 iDemand._destination,
00236                 currentDate,
00237                 iDemand._prefCabin);
00238
00239             // DEBUG
00240             // STDAIR_LOG_DEBUG ("Demand stream key: " <<
00241                 lDemandStreamKey.describe());
00242
00243             //
00244             const DemandDistribution lDemandDistribution (iDemand._demandMean,
00245                 iDemand._demandStdDev);
00246
00247             // Seed
00248             const stdair::RandomSeed_T& lRequestDateTimeSeed =
00249                 generateSeed (ioSharedGenerator);
00250             const stdair::RandomSeed_T& lDemandCharacteristicsSeed =
00251                 generateSeed (ioSharedGenerator);
00252
00253             // Delegate the call to the dedicated command
00254             DemandStream& lDemandStream =
00255                 createDemandStream (ioEventQueue, lDemandStreamKey,
00256                     iDemand._dtdProbDist, iDemand._posProbDist,
00257                     iDemand._channelProbDist,
00258                     iDemand._tripProbDist,
00259                     iDemand._stayProbDist, iDemand._ffProbDist,
00260                     iDemand._prefDepTimeProbDist,
00261                     iDemand._minWTP,
00262                     iDemand._timeValueProbDist,
00263                     lDemandDistribution, lSharedGenerator,
00264                     lRequestDateTimeSeed,
00265                     lDemandCharacteristicsSeed,
00266                     iPOSProbMass);
00267
00268             // Calculate the expected total number of events for the current
00269             // demand stream
00270             const stdair::NbOfRequests_T& lExpectedTotalNbOfEvents =
00271                 lDemandStream.getMeanNumberOfRequests();
00272
00273             ioEventQueue.addStatus (stdair::EventType::BKG_REQ,
00274                 lExpectedTotalNbOfEvents);
00275
00276

```

```

00277     }
00278     }
00279 }
00280
00281 ///////////////////////////////////////////////////////////////////
00282 stdair::RandomSeed_T DemandManager::
00283 generateSeed (stdair::RandomGeneration& ioSharedGenerator) {
00284     stdair::RealNumber_T lVariateUnif = ioSharedGenerator() * 1e9;
00285     stdair::RandomSeed_T oSeed = static_cast<stdair::RandomSeed_T>(lVariateUnif
);
00286     return oSeed;
00287 }
00288
00289 ///////////////////////////////////////////////////////////////////
00290 const bool DemandManager::
00291 stillHavingRequestsToBeGenerated (const stdair::EventQueue& iEventQueue,
00292                                   const stdair::DemandStreamKeyStr_T& iKey,
00293                                   stdair::ProgressStatusSet& ioPSS,
00294                                   const stdair::DemandGenerationMethod&
iDemandGenerationMethod) {
00295     // Retrieve the DemandStream which corresponds to the given key.
00296     const DemandStream& lDemandStream =
00297         stdair::BomManager::getObject<DemandStream> (iEventQueue, iKey);
00298
00299     // Retrieve the progress status of the demand stream.
00300     stdair::ProgressStatus
00301         lProgressStatus (lDemandStream.getNumberOfRequestsGeneratedSoFar(),
00302                         lDemandStream.getMeanNumberOfRequests(),
00303                         lDemandStream.getTotalNumberOfRequestsToBeGenerated());
00304     ioPSS.setSpecificGeneratorStatus (lProgressStatus, iKey);
00305
00306     return lDemandStream.stillHavingRequestsToBeGenerated (
iDemandGenerationMethod);
00307 }
00308
00309 ///////////////////////////////////////////////////////////////////
00310 stdair::BookingRequestPtr_T DemandManager::
00311 generateNextRequest (stdair::EventQueue& ioEventQueue,
00312                    stdair::RandomGeneration& ioGenerator,
00313                    const stdair::DemandStreamKeyStr_T& iKey,
00314                    const stdair::DemandGenerationMethod&
iDemandGenerationMethod) {
00315     // Retrieve the DemandStream which corresponds to the given key.
00316     DemandStream& lDemandStream =
00317         stdair::BomManager::getObject<DemandStream> (ioEventQueue, iKey);
00318
00319     // Generate the next booking request
00320     stdair::BookingRequestPtr_T lBookingRequest =
00321         lDemandStream.generateNextRequest (ioGenerator,
00322                                           iDemandGenerationMethod);
00323
00324     // Create an event structure
00325     stdair::EventStruct lEventStruct (stdair::EventType::BKG_REQ,
00326                                     lBookingRequest);
00327
00328     ioEventQueue.addEvent (lEventStruct);
00329
00330     return lBookingRequest;
00331 }
00332
00333 ///////////////////////////////////////////////////////////////////
00341 stdair::Count_T DemandManager::
00342 generateFirstRequests (stdair::EventQueue& ioEventQueue,
00343                      stdair::RandomGeneration& ioGenerator,
00344                      const stdair::DemandGenerationMethod&
iDemandGenerationMethod) {
00345     // Actual total number of events to be generated
00346     stdair::NbOfRequests_T lActualTotalNbOfEvents = 0.0;
00347
00348     // Retrieve the DemandStream list
00349     const DemandStreamList_T& lDemandStreamList =
00350         stdair::BomManager::getList<DemandStream> (ioEventQueue);
00351
00352     for (DemandStreamList_T::const_iterator itDemandStream =
00353          lDemandStreamList.begin();
00354          itDemandStream != lDemandStreamList.end(); ++itDemandStream) {
00355         DemandStream* lDemandStream_ptr = *itDemandStream;
00356         assert (lDemandStream_ptr != NULL);
00357
00358         lDemandStream_ptr->setBoolFirstDateTimeRequest(true);
00359
00360         // Calculate the expected total number of events for the current
00361         // demand stream
00362         const stdair::NbOfRequests_T& lActualNbOfEvents =
00363             lDemandStream_ptr->getTotalNumberOfRequestsToBeGenerated();

```



```

00366         lActualTotalNbOfEvents += lActualNbOfEvents;
00367
00368         // Retrieve the key of the demand stream
00369         const DemandStreamKey& lKey = lDemandStream_ptr->getKey();
00370
00371         // Update the progress status for the given event type (i.e.,
00372         // booking request)
00373         ioEventQueue.updateStatus (stdair::EventType::BKG_REQ, lActualNbOfEvents)
00374 ;
00375
00376         // Check whether there are still booking requests to be generated
00377         const bool stillHavingRequestsToBeGenerated =
00378         lDemandStream_ptr->stillHavingRequestsToBeGenerated (
00379         iDemandGenerationMethod);
00380
00381         if (stillHavingRequestsToBeGenerated) {
00382             // Generate the next event (booking request), and insert it
00383             // into the event queue
00384             generateNextRequest (ioEventQueue, ioGenerator, lKey.toString(),
00385                                 iDemandGenerationMethod);
00386         }
00387
00388         // Update the actual total number of events to be generated
00389         ioEventQueue.setActualTotalNbOfEvents (lActualTotalNbOfEvents);
00390
00391         // Retrieve the actual total number of events to be generated
00392         const stdair::Count_T oTotalNbOfEvents = std::floor (lActualTotalNbOfEvents
00393 );
00394
00395         //
00396         return oTotalNbOfEvents;
00397     }
00398
00399     // //////////////////////////////////////
00400     void DemandManager::reset (stdair::EventQueue& ioEventQueue,
00401                               stdair::BaseGenerator_T& ioShareGenerator) {
00402
00403         // TODO: check whether it is really necessary to destroy the
00404         // objects manually. Indeed, FacSupervisor::cleanAll() should
00405         // destroy any BOM object.
00406
00407         // Reset all the DemandStream objects
00408         const DemandStreamList_T& lDemandStreamList =
00409         stdair::BomManager::getList<DemandStream> (ioEventQueue);
00410         for (DemandStreamList_T::const_iterator itDS = lDemandStreamList.begin();
00411             itDS != lDemandStreamList.end(); ++itDS) {
00412             DemandStream* lCurrentDS_ptr = *itDS;
00413             assert (lCurrentDS_ptr != NULL);
00414             lCurrentDS_ptr->reset (ioShareGenerator);
00415         }
00416
00417         ioEventQueue.reset();
00418     }
00419
00420     // //////////////////////////////////////
00421     bool DemandManager::
00422     generateCancellation (stdair::EventQueue& ioEventQueue,
00423                           stdair::RandomGeneration& ioGenerator,
00424                           const stdair::TravelSolutionStruct& iTravelSolution,
00425                           const stdair::PartySize_T& iPartySize,
00426                           const stdair::DateTime_T& iRequestTime,
00427                           const stdair::Date_T& iDepartureDate) {
00428
00429         // Draw a random number to decide if we generate a
00430         // cancellation. For instance, the probability will be hardcoded.
00431         // The cancellation time will be generated uniformly.
00432         double lRandomNumber = ioGenerator();
00433
00434         if (lRandomNumber >= 0.5) {
00435             return false;
00436         }
00437         lRandomNumber /= 0.5;
00438
00439         // Hardcode the latest cancellation time.
00440         const stdair::Time_T lMidNight =
00441         boost::posix_time::hours (0);
00442         const stdair::DateTime_T lDepartureDateTime =
00443         boost::posix_time::ptime (iDepartureDate, lMidNight);
00444
00445         // Time to departure.
00446         const stdair::Duration_T lTimeToDeparture = lDepartureDateTime-iRequestTime
00447 ;
00448
00449         // Cancellation time to departure

```

```

00456     const long lTimeToDepartureInSeconds = lTimeToDeparture.total_seconds();
00457     const long lCancellationTimeToDepartureInSeconds =
00458         static_cast<long> (lTimeToDepartureInSeconds * lRandomNumber);
00459     const stdair::Duration_T lCancellationTimeToDeparture (0, 0,
lCancellationTimeToDepartureInSeconds);
00460
00461     // Cancellation time
00462     const stdair::DateTime_T lCancellationTime =
00463         lDepartureDateTime - lCancellationTimeToDeparture;
00464
00465     // Retrieve the segment path
00466     const stdair::SegmentPath_T lSegmentPath = iTravelSolution.getSegmentPath()
;
00467
00468     // Hardcoded class path
00469     const stdair::FareOptionStruct& lChosenFareOption =
00470         iTravelSolution.getChosenFareOption ();
00471     const stdair::ClassList_StringList_T& lClassPath =
00472         lChosenFareOption.getClassPath();
00473     std::ostringstream oStr;
00474     for (stdair::ClassList_StringList_T::const_iterator itClassList =
00475         lClassPath.begin(); itClassList != lClassPath.end(); ++itClassList)
{
00476         const stdair::ClassList_String_T& lClassList = *itClassList;
00477         assert (lClassList.size() > 0);
00478         oStr << lClassList.at(0);
00479     }
00480     const stdair::ClassList_String_T lClassList_String = oStr.str();
00481
00482     // Create the cancellation.
00483     stdair::CancellationPtr_T lCancellation_ptr =
00484         stdair::CancellationPtr_T
00485             (new stdair::CancellationStruct (lSegmentPath, lClassList_String,
00486                 iPartySize, lCancellationTime));
00487
00488     // Create an event structure
00489     stdair::EventStruct lEventStruct (stdair::EventType::CX, lCancellation_ptr)
;
00490
00491     ioEventQueue.addEvent (lEventStruct);
00492
00493     return true;
00494 }
00495
00496 // //////////////////////////////////////
00497 void DemandManager::
00498 buildSampleBom (stdair::EventQueue& ioEventQueue,
00499     stdair::RandomGeneration& ioSharedGenerator,
00500     const POSProbabilityMass_T& iPOSProbMass) {
00501
00502     //
00503     ArrivalPatternCumulativeDistribution_T lDTPProbDist;
00504     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-330
00505
00506         0));
00507     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-150
00508
00509         0.1)
00510 );
00511     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-92,
00512         0.2)
00513 );
00514     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-55,
00515         0.3)
00516 );
00517     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-34,
00518         0.4)
00519 );
00520     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-21,
00521         0.5)
00522 );
00523     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-12,
00524         0.6)
00525 );
00526     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-6,
00527         0.7)
00528 );
00529     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-3,
00530         0.8)
00531 );
00532     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(-1,
00533         0.9)
00534 );
00535     lDTPProbDist.insert (ArrivalPatternCumulativeDistribution_T::value_type(0,
00536         1.0)
00537 );
00538 }

```

```

00534 //
00535 ChannelProbabilityMassFunction_T lChannelProbDist;
00536 lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("DF"
00537 ,
00538 0.0)
00539 );
00540 lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("DN"
00541 ,
00542 0.0)
00543 );
00544 lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("IF"
00545 ,
00546 0.0)
00547 );
00548 lChannelProbDist.insert (ChannelProbabilityMassFunction_T::value_type ("IN"
00549 ,
00550 1.0)
00551 );
00552 //
00553 TripTypeProbabilityMassFunction_T lTripProbDist;
00554 lTripProbDist.insert (TripTypeProbabilityMassFunction_T::value_type ("RO",
00555 0.0));
00556 lTripProbDist.insert (TripTypeProbabilityMassFunction_T::value_type ("RI",
00557 0.0));
00558 lTripProbDist.insert (TripTypeProbabilityMassFunction_T::value_type ("OW",
00559 1.0));
00560 //
00561 StayDurationProbabilityMassFunction_T lStayProbDist;
00562 lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (0,
00563 0.1)
00564 );
00565 lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (1,
00566 0.1)
00567 );
00568 lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (2,
00569 0.15)
00570 );
00571 lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (3,
00572 0.15)
00573 );
00574 lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (4,
00575 0.15)
00576 );
00577 lStayProbDist.insert (StayDurationProbabilityMassFunction_T::value_type (5,
00578 0.35)
00579 );
00580 //
00581 FrequentFlyerProbabilityMassFunction_T lFFProbDist;
00582 lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type ("P",
00583 0.1)
00584 );
00585 lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type ("G",
00586 0.01)
00587 );
00588 lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type ("S",
00589 0.09)
00590 );
00591 lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type ("M",
00592 0.4)
00593 );
00594 lFFProbDist.insert (FrequentFlyerProbabilityMassFunction_T::value_type ("N",
00595 0.4)
00596 );
00597 //
00598 ValueOfTimeContinuousDistribution_T lTimeValueProbDist;
00599 lTimeValueProbDist.insert (ValueOfTimeContinuousDistribution_T::value_type (1
00600 ,
00601 0
00602 ));
00603 lTimeValueProbDist.insert (ValueOfTimeContinuousDistribution_T::value_type (6
00604 ,
00605 0
00606 ));
00607 lTimeValueProbDist.insert (ValueOfTimeContinuousDistribution_T::value_type (1
00608 ,
00609 1
00610 ));
00611 //
00612 /*
00613 =====*/
00614 // Key of the demand stream
00615 const stdair::AirportCode_T lSINOrigin ("SIN");
00616 const stdair::AirportCode_T lBKKDestination ("BKK");
00617 const stdair::Date_T lDepDate (2010, 2, 8);
00618 const stdair::CabinCode_T lCabin ("Y");
00619

```

```

00597     //
00598     const DemandStreamKey lSINBKKDemandStreamKey (lSINOrigin, lBKKDestination,
lDepDate,
00599                                     lCabin);
00600
00601     // DEBUG
00602     // STDAIR_LOG_DEBUG ("Demand stream key: " << lDemandStreamKey.describe());
00603
00604     // Distribution for the number of requests
00605     const stdair::MeanValue_T lSINBKKDemandMean (60.0);
00606     const stdair::StdDevValue_T lSINBKKDemandStdDev (4.0);
00607     const DemandDistribution lSINBKKDemandDistribution (lSINBKKDemandMean,
lSINBKKDemandStdDev);
00608
00609     // Seed
00610     const stdair::RandomSeed_T& lSINBKKRequestDateTimeSeed =
generateSeed (ioSharedGenerator);
00611     const stdair::RandomSeed_T& lSINBKKDemandCharacteristicsSeed =
generateSeed (ioSharedGenerator);
00612
00613     //
00614     POSProbabilityMassFunction_T lSINBKKPOSProbDist;
00615     lSINBKKPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("SIN",
1.0));
00616     lSINBKKPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("BKK",
0.0));
00617
00618     //
00619     PreferredDepartureTimeContinuousDistribution_T lSINPrefDepTimeProbDist;
00620     lSINPrefDepTimeProbDist.
insert (PreferredDepartureTimeContinuousDistribution_T::value_type (6, 0)
);
00621     lSINPrefDepTimeProbDist.
insert (PreferredDepartureTimeContinuousDistribution_T::value_type (8,
0.7)
);
00622     lSINPrefDepTimeProbDist.
insert (PreferredDepartureTimeContinuousDistribution_T::value_type (10,
0.8)
);
00623     lSINPrefDepTimeProbDist.
insert (PreferredDepartureTimeContinuousDistribution_T::value_type (12,
0.9)
);
00624     lSINPrefDepTimeProbDist.
insert (PreferredDepartureTimeContinuousDistribution_T::value_type (14,
1.0)
);
00625
00626     //
00627     const stdair::WTP_T lSINBKKWTP (400.0);
00628
00629     // Delegate the call to the dedicated command
00630     DemandStream& lSINBKKDemandStream =
createDemandStream (ioEventQueue, lSINBKKDemandStreamKey, lDTDProbDist,
lSINBKKPOSProbDist, lChannelProbDist, lTripProbDist,
lStayProbDist, lFFPProbDist, lSINPrefDepTimeProbDist,
lSINBKKWTP, lTimeValueProbDist,
lSINBKKDemandDistribution,
ioSharedGenerator.getBaseGenerator(),
lSINBKKRequestDateTimeSeed,
lSINBKKDemandCharacteristicsSeed, iPOSProbMass);
00631
00632     // Calculate the expected total number of events for the current
00633     // demand stream
00634     const stdair::NbOfRequests_T& lSINBKKExpectedNbOfEvents =
lSINBKKDemandStream.getMeanNumberOfRequests();
00635
00636     /*
=====*/
00637
00638     // Key of the demand stream
00639     const stdair::AirportCode_T lBKKOrigin ("BKK");
00640     const stdair::AirportCode_T lHKGDestination ("HKG");
00641
00642     //
00643     const DemandStreamKey lBKKHKGDemandStreamKey (lBKKOrigin, lHKGDestination,
lDepDate,
00644                                     lCabin);
00645
00646     // DEBUG
00647     // STDAIR_LOG_DEBUG ("Demand stream key: " << lDemandStreamKey.describe());
00648
00649     // Distribution for the number of requests
00650     const stdair::MeanValue_T lBKKHKGDemandMean (60.0);

```

```

00672     const stdair::StdDevValue_T lBKKHKGDemandStdDev (4.0);
00673     const DemandDistribution lBKKHKGDemandDistribution (lBKKHKGDemandMean,
lBKKHKGDemandStdDev);
00674
00675     // Seed
00676     const stdair::RandomSeed_T& lBKKHKGRequestDateTimeSeed =
00677         generateSeed (ioSharedGenerator);
00678     const stdair::RandomSeed_T& lBKKHKGDemandCharacteristicsSeed =
00679         generateSeed (ioSharedGenerator);
00680
00681
00682     //
00683     POSProbabilityMassFunction_T lBKKHKGPOSProbDist;
00684     lBKKHKGPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("BKK",
1.0));
00685     lBKKHKGPOSProbDist.insert (POSProbabilityMassFunction_T::value_type ("HKG",
0.0));
00686
00687     //
00688     PreferredDepartureTimeContinuousDistribution_T lBKKPrefDepTimeProbDist;
00689     lBKKPrefDepTimeProbDist.
00690         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (8, 0)
);
00691     lBKKPrefDepTimeProbDist.
00692         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (10,
0.2))
00693 ;
00694     lBKKPrefDepTimeProbDist.
00695         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (1,
0.6))
00696 ;
00697     lBKKPrefDepTimeProbDist.
00698         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (14,
0.8))
00699 ;
00700     lBKKPrefDepTimeProbDist.
00701         insert (PreferredDepartureTimeContinuousDistribution_T::value_type (16,
1.0))
00702 ;
00703
00704     //
00705     const stdair::WTP_T lBKKHKGWTP (400.0);
00706
00707
00708     // Delegate the call to the dedicated command
00709     DemandStream& lBKKHKGDemandStream =
00710         createDemandStream (ioEventQueue, lBKKHKGDemandStreamKey, lDTPProbDist,
lBKKHKGPOSProbDist, lChannelProbDist, lTripProbDist,
00711             lStayProbDist, lFFProbDist, lBKKPrefDepTimeProbDist,
lBKKHKGWTP, lTimeValueProbDist,
00712             lBKKHKGDemandDistribution,
00713             ioSharedGenerator.getBaseGenerator(),
00714             lBKKHKGRequestDateTimeSeed,
00715             lBKKHKGDemandCharacteristicsSeed, iPOSProbMass);
00716
00717
00718     // Calculate the expected total number of events for the current
00719     // demand stream
00720     const stdair::NbOfRequests_T& lBKKHKGExpectedNbOfEvents =
00721         lBKKHKGDemandStream.getMeanNumberOfRequests();
00722
00723     /*
=====*/
00724
00725     // Key of the demand stream
00726
00727     //
00728     const DemandStreamKey lSINHKGDemandStreamKey (lSINOrigin, lHKGDestination,
lDepDate,
00729         lCabin);
00730
00731     // DEBUG
00732     // STDAIR_LOG_DEBUG ("Demand stream key: " << lDemandStreamKey.describe());
00733
00734     // Distribution for the number of requests
00735     const stdair::MeanValue_T lSINHKGDemandMean (60.0);
00736     const stdair::StdDevValue_T lSINHKGDemandStdDev (4.0);
00737     const DemandDistribution lSINHKGDemandDistribution (lSINHKGDemandMean,
lSINHKGDemandStdDev);
00738
00739     // Seed
00740     const stdair::RandomSeed_T& lSINHKGRequestDateTimeSeed =
00741         generateSeed (ioSharedGenerator);
00742     const stdair::RandomSeed_T& lSINHKGDemandCharacteristicsSeed =
00743         generateSeed (ioSharedGenerator);
00744
00745
00746     //

```

```

00747     POSProbabilityMassFunction_T lSINHKGPOSProbDist;
00748     lSINHKGPOSProbDist.insert (POSPProbabilityMassFunction_T::value_type ("SIN",
00749     1.0));
00749     lSINHKGPOSProbDist.insert (POSPProbabilityMassFunction_T::value_type ("HKG",
00750     0.0));
00750
00751     //
00752     const stdair::WTP_T lSINHKGWTP (750.0);
00753
00754     // Delegate the call to the dedicated command
00755     DemandStream& lSINHKGDemandStream =
00756         createDemandStream (ioEventQueue, lSINHKGDemandStreamKey, lDTPProbDist,
00757                             lSINHKGPOSProbDist, lChannelProbDist, lTripProbDist,
00758                             lStayProbDist, lFFPProbDist, lSINPrefDepTimeProbDist,
00759                             lSINHKGWTP, lTimeValueProbDist,
00760                             lSINHKGDemandDistribution,
00761                             ioSharedGenerator.getBaseGenerator(),
00762                             lSINHKGRequestDateTimeSeed,
00763                             lSINHKGDemandCharacteristicsSeed, iPOSProbMass);
00764
00765     // Calculate the expected total number of events for the current
00766     // demand stream
00767     const stdair::NbOfRequests_T& lSINHKGExpectedNbOfEvents =
00768         lSINHKGDemandStream.getMeanNumberOfRequests();
00769
00770     /*
00771     =====*/
00775     const stdair::NbOfRequests_T lExpectedTotalNbOfEvents =
00776         lSINBKKEExpectedNbOfEvents + lBKKHKGExpectedNbOfEvents +
00777         lSINHKGExpectedNbOfEvents;
00777     ioEventQueue.addStatus (stdair::EventType::BKG_REQ,
00778                             lExpectedTotalNbOfEvents);
00779 }
00780
00781 }

```

25.85 trademgen/command/DemandManager.hpp File Reference

```

#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/RandomGeneration.hpp>
#include <stdair/basic/DemandGenerationMethod.hpp>
#include <stdair/bom/BookingRequestTypes.hpp>
#include <stdair/command/CmdAbstract.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
#include <trademgen/bom/DemandStreamKey.hpp>

```

Classes

- class [TRADEMGEN::DemandManager](#)
Utility class for Demand and [DemandStream](#) objects.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)
- namespace [TRADEMGEN::DemandParserHelper](#)

25.86 DemandManager.hpp

```

00001 #ifndef __TRADEMGEN_CMD_DEMANDMANAGER_HPP
00002 #define __TRADEMGEN_CMD_DEMANDMANAGER_HPP
00003
00004 // =====

```

```

00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/basic/RandomGeneration.hpp>
00010 #include <stdair/basic/DemandGenerationMethod.hpp>
00011 #include <stdair/bom/BookingRequestTypes.hpp>
00012 #include <stdair/command/CmdAbstract.hpp>
00013 // TraDemGen
00014 #include <trademgen/TRADEMGEN_Types.hpp>
00015 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00016 #include <trademgen/bom/DemandStreamKey.hpp>
00017
00018 // Forward declarations
00019 namespace stdair {
00020     class EventQueue;
00021     struct ProgressStatusSet;
00022     struct TravelSolutionStruct;
00023 }
00024
00025 namespace TRADEMGEN {
00026
00027     // Forward declarations
00028     struct DemandDistribution;
00029     struct DemandStruct;
00030     class DemandStream;
00031     namespace DemandParserHelper {
00032         struct doEndDemand;
00033     }
00034
00038     class DemandManager : public stdair::CmdAbstract {
00039     friend struct DemandParserHelper::doEndDemand;
00040     friend class TRADEMGEN_Service;
00041
00042     private:
00043         // ////////// Business methodes //////////
00123         static void buildSampleBomStd (stdair::EventQueue&,
stdair::RandomGeneration&,
00124                                     const POSProbabilityMass_T&);
00125
00126         // Demand sample bom for partnerships study.
00127         static void buildSampleBom (stdair::EventQueue&, stdair::RandomGeneration&,
00128                                     const POSProbabilityMass_T&);
00129
00137         static void createDemandCharacteristics (stdair::EventQueue&,
stdair::RandomGeneration&,
00138                                                 const POSProbabilityMass_T&,
00139                                                 const DemandStruct&);
00140
00149         static stdair::RandomSeed_T generateSeed (stdair::RandomGeneration&);
00150
00172         static DemandStream&
createDemandStream (stdair::EventQueue&,
00173                     const DemandStreamKey&,
00174                     const ArrivalPatternCumulativeDistribution_T&,
00175                     const POSProbabilityMassFunction_T&,
00176                     const ChannelProbabilityMassFunction_T&,
00177                     const TripTypeProbabilityMassFunction_T&,
00178                     const StayDurationProbabilityMassFunction_T&,
00179                     const FrequentFlyerProbabilityMassFunction_T&,
00180                     const PreferredDepartureTimeContinuousDistribution_T&,
00181                     const stdair::WTP_T&,
00182                     const ValueOfTimeContinuousDistribution_T&,
00183                     const DemandDistribution&, stdair::BaseGenerator_T&,
00184                     const stdair::RandomSeed_T&,
00185                     const stdair::RandomSeed_T&,
00186                     const POSProbabilityMass_T&);
00187
00205         static const bool
stillHavingRequestsToBeGenerated (const stdair::EventQueue&,
00206                                   const stdair::DemandStreamKeyStr_T&,
00207                                   stdair::ProgressStatusSet&,
00208                                   const stdair::DemandGenerationMethod&);
00209
00225         static stdair::Count_T generateFirstRequests (stdair::EventQueue&,
stdair::RandomGeneration&,
00226                                                         const
stdair::DemandGenerationMethod&);
00227
00228
00249         static stdair::BookingRequestPtr_T
generateNextRequest (stdair::EventQueue&, stdair::RandomGeneration&,
00250                     const stdair::DemandStreamKeyStr_T&,
00251                     const stdair::DemandGenerationMethod&);
00252
00262         static void reset (stdair::EventQueue&, stdair::BaseGenerator_T&);
00263
00267         static bool generateCancellation (stdair::EventQueue&,

```

```

00268                                     stdair::RandomGeneration&,
00269                                     const stdair::TravelSolutionStruct&,
00270                                     const stdair::PartySize_T&,
00271                                     const stdair::DateTime_T&,
00272                                     const stdair::Date_T&);
00273     };
00274
00275 }
00276 #endif // __TRADEMGEN_CMD_DEMANDMANAGER_HPP

```

25.87 trademgen/command/DemandParser.cpp File Reference

```

#include <cassert>
#include <stdair/basic/BasFileMgr.hpp>
#include <stdair/basic/RandomGeneration.hpp>
#include <stdair/bom/Inventory.hpp>
#include <trademgen/command/DemandParserHelper.hpp>
#include <trademgen/command/DemandParser.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.88 DemandParser.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // StdAir
00007 #include <stdair/basic/BasFileMgr.hpp>
00008 #include <stdair/basic/RandomGeneration.hpp>
00009 #include <stdair/bom/Inventory.hpp>
00010 // TraDemGen
00011 #include <trademgen/command/DemandParserHelper.hpp>
00012 #include <trademgen/command/DemandParser.hpp>
00013
00014 namespace TRADEMGEN {
00015
00016 // //////////////////////////////////////
00017 void DemandParser::
00018 generateDemand (const stdair::Filename_T& iFilename,
00019                 stdair::EventQueue& ioEventQueue,
00020                 stdair::RandomGeneration& ioSharedGenerator,
00021                 const POSProbabilityMass_T& iDefaultPOSProbabilityMass) {
00022
00023     // Check that the file path given as input corresponds to an actual file
00024     const bool doesExistAndIsReadable =
00025         stdair::BasFileMgr::doesExistAndIsReadable (iFilename);
00026     if (doesExistAndIsReadable == false) {
00027         STDAIR_LOG_ERROR ("The demand input file '" << iFilename
00028                          << "' does not exist or can not be read");
00029
00030         throw DemandInputFileNotFoundException ("The demand file '" + iFilename
00031                                                  + "' does not exist or can not "
00032                                                  "be read");
00033     }
00034
00035     // Initialise the demand file parser.
00036     DemandFileParser lDemandParser (ioEventQueue, ioSharedGenerator,
00037                                     iDefaultPOSProbabilityMass, iFilename);
00038
00039     // Parse the CSV-formatted demand input file, and generate the
00040     // corresponding DemandCharacteristic objects.
00041     lDemandParser.generateDemand();
00042 }
00043
00044 }

```


25.89 trademgen/command/DemandParser.hpp File Reference

```
#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/command/CmdAbstract.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
```

Classes

- class [TRADEMGEN::DemandParser](#)
Class wrapping the parser entry point.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)

25.90 DemandParser.hpp

```
00001 #ifndef __TRADEMGEN_CMD_DEMANDPARSER_HPP
00002 #define __TRADEMGEN_CMD_DEMANDPARSER_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_basic_types.hpp>
00011 #include <stdair/command/CmdAbstract.hpp>
00012 // TraDemGen
00013 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00014
00016 namespace stdair {
00017     class EventQueue;
00018     struct RandomGeneration;
00019 }
00020
00021 namespace TRADEMGEN {
00022
00026     class DemandParser : public stdair::CmdAbstract {
00027     public:
00040         static void generateDemand (const stdair::Filename_T&, stdair::EventQueue&,
00041                                     stdair::RandomGeneration&,
00042                                     const POSProbabilityMass_T&);
00043     };
00044 }
00045 #endif // __TRADEMGEN_CMD_DEMANDPARSER_HPP
```

25.91 trademgen/command/DemandParserHelper.cpp File Reference

```
#include <cassert>
#include <stdair/basic/RandomGeneration.hpp>
#include <stdair/basic/BasFileMgr.hpp>
#include <stdair/bom/EventQueue.hpp>
#include <stdair/service/Logger.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>
#include <trademgen/command/DemandParserHelper.hpp>
#include <trademgen/command/DemandManager.hpp>
```

Namespaces

- namespace [TRADEMGEN](#)
- namespace [TRADEMGEN::DemandParserHelper](#)

Functions

- repeat_p_t [TRADEMGEN::DemandParserHelper::airline_code_p](#) (chset_t("0-9A-Z").derived(), 2, 3)
- bounded1_4_p_t [TRADEMGEN::DemandParserHelper::flight_number_p](#) (uint1_4_p.derived(), 0u, 9999u)
- bounded4_p_t [TRADEMGEN::DemandParserHelper::year_p](#) (uint4_p.derived(), 2000u, 2099u)
- bounded2_p_t [TRADEMGEN::DemandParserHelper::month_p](#) (uint2_p.derived(), 1u, 12u)
- bounded2_p_t [TRADEMGEN::DemandParserHelper::day_p](#) (uint2_p.derived(), 1u, 31u)
- repeat_p_t [TRADEMGEN::DemandParserHelper::dow_p](#) (chset_t("0-1").derived().derived(), 7, 7)
- repeat_p_t [TRADEMGEN::DemandParserHelper::airport_p](#) (chset_t("0-9A-Z").derived(), 3, 3)
- bounded1_2_p_t [TRADEMGEN::DemandParserHelper::hours_p](#) (uint1_2_p.derived(), 0u, 23u)
- bounded2_p_t [TRADEMGEN::DemandParserHelper::minutes_p](#) (uint2_p.derived(), 0u, 59u)
- bounded2_p_t [TRADEMGEN::DemandParserHelper::seconds_p](#) (uint2_p.derived(), 0u, 59u)
- chset_t [TRADEMGEN::DemandParserHelper::cabin_code_p](#) ("A-Z")
- chset_t [TRADEMGEN::DemandParserHelper::passenger_type_p](#) ("A-Z")
- chset_t [TRADEMGEN::DemandParserHelper::ff_type_p](#) ("A-Z")
- repeat_p_t [TRADEMGEN::DemandParserHelper::class_code_list_p](#) (chset_t("A-Z").derived(), 1, 26)
- bounded1_3_p_t [TRADEMGEN::DemandParserHelper::stay_duration_p](#) (uint1_3_p.derived(), 0u, 999u)

Variables

- int1_p_t [TRADEMGEN::DemandParserHelper::int1_p](#)
- uint2_p_t [TRADEMGEN::DemandParserHelper::uint2_p](#)
- uint1_2_p_t [TRADEMGEN::DemandParserHelper::uint1_2_p](#)
- uint1_3_p_t [TRADEMGEN::DemandParserHelper::uint1_3_p](#)
- uint4_p_t [TRADEMGEN::DemandParserHelper::uint4_p](#)
- uint1_4_p_t [TRADEMGEN::DemandParserHelper::uint1_4_p](#)
- int1_p_t [TRADEMGEN::DemandParserHelper::family_code_p](#)

25.92 DemandParserHelper.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // StdAir
00007 #include <stdair/basic/RandomGeneration.hpp>
00008 #include <stdair/basic/BasFileMgr.hpp>
00009 #include <stdair/bom/EventQueue.hpp>
00010 #include <stdair/service/Logger.hpp>
00011 // TraDemGen
00012 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00013 // #define BOOST_SPIRIT_DEBUG
00014 #include <trademgen/command/DemandParserHelper.hpp>
00015 #include <trademgen/command/DemandManager.hpp>
00016
00017 namespace bsc = boost::spirit::classic;
00018
00019 namespace TRADEMGEN {
00020
00021     namespace DemandParserHelper {
00022
00023         // //////////////////////////////////////
00024         // Semantic actions
00025         // //////////////////////////////////////
00026
00027         ParserSemanticAction::ParserSemanticAction (DemandStruct& ioDemand)
00028             : _demand (ioDemand) {
00029         }
00030

```

```

00031 // //////////////////////////////////////
00032 storePrefDepDateRangeStart::
00033 storePrefDepDateRangeStart (DemandStruct& ioDemand)
00034 : ParserSemanticAction (ioDemand) {
00035 }
00036
00037 // //////////////////////////////////////
00038 void storePrefDepDateRangeStart::operator() (iterator_t iStr,
00039 iterator_t iStrEnd) const {
00040     _demand._prefDepDateStart = _demand.getDate();
00041
00042     // Reset the number of seconds
00043     _demand._itSeconds = 0;
00044 }
00045
00046 // //////////////////////////////////////
00047 storePrefDepDateRangeEnd::
00048 storePrefDepDateRangeEnd (DemandStruct& ioDemand)
00049 : ParserSemanticAction (ioDemand) {
00050 }
00051
00052 // //////////////////////////////////////
00053 void storePrefDepDateRangeEnd::operator() (iterator_t iStr,
00054 iterator_t iStrEnd) const {
00055     // As a Boost date period (DatePeriod_T) defines the last day of
00056     // the period to be end-date - one day, we have to add one day to that
00057     // end date before.
00058     const stdair::DateOffset_T oneDay (1);
00059     _demand._prefDepDateEnd = _demand.getDate() + oneDay;
00060
00061     // Transform the date pair (i.e., the date range) into a date period
00062     _demand._dateRange =
00063         stdair::DatePeriod_T (_demand._prefDepDateStart,
00064             _demand._prefDepDateEnd);
00065
00066     // Reset the number of seconds
00067     _demand._itSeconds = 0;
00068 }
00069
00070 // //////////////////////////////////////
00071 storeDow::storeDow (DemandStruct& ioDemand)
00072 : ParserSemanticAction (ioDemand) {
00073 }
00074
00075 // //////////////////////////////////////
00076 void storeDow::operator() (iterator_t iStr, iterator_t iStrEnd) const {
00077     stdair::DOW_String_T lDow (iStr, iStrEnd);
00078     _demand._dow = lDow;
00079 }
00080
00081 // //////////////////////////////////////
00082 storeOrigin::storeOrigin (DemandStruct& ioDemand)
00083 : ParserSemanticAction (ioDemand) {
00084 }
00085
00086 // //////////////////////////////////////
00087 void storeOrigin::operator() (iterator_t iStr, iterator_t iStrEnd) const {
00088     stdair::AirportCode_T lOrigin (iStr, iStrEnd);
00089     _demand._origin = lOrigin;
00090 }
00091
00092 // //////////////////////////////////////
00093 storeDestination::storeDestination (DemandStruct& ioDemand)
00094 : ParserSemanticAction (ioDemand) {
00095 }
00096
00097 // //////////////////////////////////////
00098 void storeDestination::operator() (iterator_t iStr,
00099 iterator_t iStrEnd) const {
00100     stdair::AirportCode_T lDestination (iStr, iStrEnd);
00101     _demand._destination = lDestination;
00102 }
00103
00104 // //////////////////////////////////////
00105 storePrefCabin::storePrefCabin (DemandStruct& ioDemand)
00106 : ParserSemanticAction (ioDemand) {
00107 }
00108
00109 // //////////////////////////////////////
00110 void storePrefCabin::operator() (iterator_t iStr,
00111 iterator_t iStrEnd) const {
00112     stdair::CabinCode_T lPrefCabin (iStr, iStrEnd);
00113     _demand._prefCabin = lPrefCabin;
00114     //STDAIR_LOG_DEBUG ("Preferred cabin: " << lPrefCabin);
00115 }
00116
00117 // //////////////////////////////////////

```

```

00118     storeDemandMean::storeDemandMean (DemandStruct& ioDemand)
00119     : ParserSemanticAction (ioDemand) {
00120     }
00121
00122     // //////////////////////////////////////
00123 void storeDemandMean::operator() (double iReal) const {
00124     _demand._demandMean = iReal;
00125     //STDAIR_LOG_DEBUG ("Demand mean: " << iReal);
00126 }
00127
00128     // //////////////////////////////////////
00129 storeDemandStdDev::storeDemandStdDev (DemandStruct& ioDemand)
00130     : ParserSemanticAction (ioDemand) {
00131     }
00132
00133     // //////////////////////////////////////
00134 void storeDemandStdDev::operator() (double iReal) const {
00135     _demand._demandStdDev = iReal;
00136     //STDAIR_LOG_DEBUG ("Demand stddev: " << iReal);
00137 }
00138
00139     // //////////////////////////////////////
00140 storePosCode::storePosCode (DemandStruct& ioDemand)
00141     : ParserSemanticAction (ioDemand) {
00142     }
00143
00144     // //////////////////////////////////////
00145 void storePosCode::operator() (iterator_t iStr, iterator_t iStrEnd) const {
00146     const stdair::AirportCode_T lPosCode (iStr, iStrEnd);
00147     _demand._itPosCode = lPosCode;
00148     //STDAIR_LOG_DEBUG ("Pos code: " << lPosCode);
00149 }
00150
00151     // //////////////////////////////////////
00152 storePosProbMass::storePosProbMass (DemandStruct& ioDemand)
00153     : ParserSemanticAction (ioDemand) {
00154     }
00155
00156     // //////////////////////////////////////
00157 void storePosProbMass::operator() (double iReal) const {
00158     const bool hasInsertBeenSuccessfull =
00159         _demand._posProbDist.
00160         insert (POSProbabilityMassFunction_T::
00161             value_type (_demand._itPosCode, iReal)).second;
00162     if (hasInsertBeenSuccessfull == false) {
00163         STDAIR_LOG_ERROR ("The same POS code ('" << _demand._itPosCode
00164             << "') has probably been given twice");
00165         throw stdair::CodeDuplicationException ("The same POS code ('"
00166             + _demand._itPosCode
00167             + "') has probably been given
00168             twice");
00169     }
00170     //STDAIR_LOG_DEBUG ("PosProbMass: " << iReal);
00171 }
00172
00173     // //////////////////////////////////////
00174 storeChannelCode::storeChannelCode (DemandStruct& ioDemand)
00175     : ParserSemanticAction (ioDemand) {
00176     }
00177
00178     // //////////////////////////////////////
00179 void storeChannelCode::operator() (iterator_t iStr,
00180     iterator_t iStrEnd) const {
00181     _demand._itChannelCode = std::string (iStr, iStrEnd);
00182     //STDAIR_LOG_DEBUG ("Channel code: " << _demand._itChannelCode);
00183 }
00184
00185     // //////////////////////////////////////
00186 storeChannelProbMass::storeChannelProbMass (DemandStruct& ioDemand)
00187     : ParserSemanticAction (ioDemand) {
00188     }
00189
00190     // //////////////////////////////////////
00191 void storeChannelProbMass::operator() (double iReal) const {
00192     const bool hasInsertBeenSuccessfull =
00193         _demand._channelProbDist.
00194         insert (ChannelProbabilityMassFunction_T::
00195             value_type (_demand._itChannelCode, iReal)).second;
00196     if (hasInsertBeenSuccessfull == false) {
00197         STDAIR_LOG_ERROR ("The same channel type code ('"
00198             << _demand._itChannelCode
00199             << "') has probably been given twice");
00200         throw stdair::CodeDuplicationException ("The same channel type code ('"
00201             + _demand._itChannelCode
00202             + "') has probably been given
00203             twice");

```

```

00203     }
00204
00205     //STDAIR_LOG_DEBUG ("ChannelProbMass: " << iReal);
00206 }
00207
00208 // //////////////////////////////////////
00209 storeTripCode::storeTripCode (DemandStruct& ioDemand)
00210 : ParserSemanticAction (ioDemand) {
00211 }
00212
00213 // //////////////////////////////////////
00214 void storeTripCode::operator() (iterator_t iStr,
00215                               iterator_t iStrEnd) const {
00216     _demand._itTripCode = std::string (iStr, iStrEnd);
00217     //STDAIR_LOG_DEBUG ("Trip code: " << _demand._itTripCode);
00218 }
00219
00220 // //////////////////////////////////////
00221 storeTripProbMass::storeTripProbMass (DemandStruct& ioDemand)
00222 : ParserSemanticAction (ioDemand) {
00223 }
00224
00225 // //////////////////////////////////////
00226 void storeTripProbMass::operator() (double iReal) const {
00227     const bool hasInsertBeenSuccessfull =
00228         _demand._tripProbDist.
00229         insert (TripTypeProbabilityMassFunction_T::
00230             value_type (_demand._itTripCode, iReal)).second;
00231     if (hasInsertBeenSuccessfull == false) {
00232         STDAIR_LOG_ERROR ("The same trip type code ('"
00233             << _demand._itTripCode
00234             << "') has probably been given twice");
00235         throw stdair::CodeDuplicationException ("The same trip type code ('"
00236             + _demand._itTripCode
00237             + "') has probably been given
00238         twice");
00239     }
00240     //STDAIR_LOG_DEBUG ("TripProbMass: " << iReal);
00241 }
00242
00243 // //////////////////////////////////////
00244 storeStayCode::storeStayCode (DemandStruct& ioDemand)
00245 : ParserSemanticAction (ioDemand) {
00246 }
00247
00248 // //////////////////////////////////////
00249 void storeStayCode::operator() (unsigned int iInteger) const {
00250     const stdair::DayDuration_T lStayDuration (iInteger);
00251     _demand._itStayDuration = lStayDuration;
00252     // STDAIR_LOG_DEBUG ("Stay duration: " << lStayDuration);
00253 }
00254
00255 // //////////////////////////////////////
00256 storeStayProbMass::storeStayProbMass (DemandStruct& ioDemand)
00257 : ParserSemanticAction (ioDemand) {
00258 }
00259
00260 // //////////////////////////////////////
00261 void storeStayProbMass::operator() (double iReal) const {
00262     const bool hasInsertBeenSuccessfull =
00263         _demand._stayProbDist.
00264         insert (StayDurationProbabilityMassFunction_T::
00265             value_type (_demand._itStayDuration, iReal)).second;
00266     if (hasInsertBeenSuccessfull == false) {
00267         std::ostream oStr;
00268         oStr << "The same stay duration ('" << _demand._itStayDuration
00269             << "') has probably been given twice";
00270         STDAIR_LOG_ERROR (oStr.str());
00271         throw stdair::CodeDuplicationException (oStr.str());
00272     }
00273     // STDAIR_LOG_DEBUG ("StayProbMass: " << iReal);
00274 }
00275
00276 // //////////////////////////////////////
00277 storeFFCode::storeFFCode (DemandStruct& ioDemand)
00278 : ParserSemanticAction (ioDemand) {
00279 }
00280
00281 // //////////////////////////////////////
00282 void storeFFCode::operator() (iterator_t iStr, iterator_t iStrEnd) const {
00283     _demand._itFFCode = std::string (iStr, iStrEnd);
00284     //STDAIR_LOG_DEBUG ("FF code: " << _demand._itFFCode);
00285 }
00286
00287 // //////////////////////////////////////
00288

```

```

00289     storeFFProbMass::storeFFProbMass (DemandStruct& ioDemand)
00290     : ParserSemanticAction (ioDemand) {
00291     }
00292
00293     // //////////////////////////////////////
00294 void storeFFProbMass::operator() (double iReal) const {
00295     const bool hasInsertBeenSuccessfull =
00296         _demand._ffProbDist.
00297         insert (FrequentFlyerProbabilityMassFunction_T::
00298             value_type (_demand._itFFCode, iReal)).second;
00299     if (hasInsertBeenSuccessfull == false) {
00300         STDAIR_LOG_ERROR ("The same Frequent Flyer code ('"
00301             << _demand._itFFCode
00302             << "') has probably been given twice");
00303         throw stdair::CodeDuplicationException("The same Frequent Flyer code ('"
00304             + _demand._itFFCode
00305             + "') has probably been given
00306             twice");
00307     }
00308     //STDAIR_LOG_DEBUG ("FfProbMass: " << iReal);
00309 }
00310
00311 // //////////////////////////////////////
00312 storePrefDepTime::storePrefDepTime (DemandStruct& ioDemand)
00313 : ParserSemanticAction (ioDemand) {
00314 }
00315
00316 // //////////////////////////////////////
00317 void storePrefDepTime::operator() (iterator_t iStr,
00318     iterator_t iStrEnd) const {
00319     _demand._itPrefDepTime = _demand.getTime();
00320
00321     // DEBUG
00322     // STDAIR_LOG_DEBUG ("Pref dep time: " << _demand._itHours << ":"
00323     // << _demand._itMinutes << ":" << _demand._itSeconds
00324     // << " ==> " << _demand._itPrefDepTime);
00325
00326     // Reset the number of minutes and seconds
00327     _demand._itMinutes = 0;
00328     _demand._itSeconds = 0;
00329 }
00330
00331 // //////////////////////////////////////
00332 storePrefDepTimeProbMass::storePrefDepTimeProbMass (DemandStruct& ioDemand)
00333 : ParserSemanticAction (ioDemand) {
00334 }
00335
00336 // //////////////////////////////////////
00337 void storePrefDepTimeProbMass::operator() (double iReal) const {
00338     const stdair::IntDuration_T lIntDuration =
00339         _demand._itPrefDepTime.total_seconds();
00340
00341     _demand._prefDepTimeProbDist.
00342     insert (PreferredDepartureTimeContinuousDistribution_T::
00343         value_type (lIntDuration, iReal));
00344     //STDAIR_LOG_DEBUG ("PrefDepTimeProbMass: " << iReal);
00345 }
00346
00347 // //////////////////////////////////////
00348 storeWTP::storeWTP (DemandStruct& ioDemand)
00349 : ParserSemanticAction (ioDemand) {
00350 }
00351
00352 // //////////////////////////////////////
00353 void storeWTP::operator() (double iReal) const {
00354     _demand._minWTP = iReal;
00355     //STDAIR_LOG_DEBUG ("WTP: " << iReal);
00356 }
00357
00358 // //////////////////////////////////////
00359 storeTimeValue::storeTimeValue (DemandStruct& ioDemand)
00360 : ParserSemanticAction (ioDemand) {
00361 }
00362
00363 // //////////////////////////////////////
00364 void storeTimeValue::operator() (double iReal) const {
00365     _demand._itTimeValue = iReal;
00366     //STDAIR_LOG_DEBUG ("Time value: " << iReal);
00367 }
00368
00369 // //////////////////////////////////////
00370 storeTimeValueProbMass::storeTimeValueProbMass (DemandStruct& ioDemand)
00371 : ParserSemanticAction (ioDemand) {
00372 }
00373

```

```

00374 // //////////////////////////////////////
00375 void storeTimeValueProbMass::operator() (double iReal) const {
00376     _demand._timeValueProbDist.
00377         insert (ValueOfTimeContinuousDistribution_T::
00378             value_type (_demand._itTimeValue, iReal));
00379     //STDAIR_LOG_DEBUG ("TimeValueProbMass: " << iReal);
00380 }
00381
00382 // //////////////////////////////////////
00383 storeDTD::storeDTD (DemandStruct& ioDemand)
00384     : ParserSemanticAction (ioDemand) {
00385 }
00386
00387 // //////////////////////////////////////
00388 void storeDTD::operator() (unsigned int iInteger) const {
00389     const stdair::DayDuration_T lDTD (iInteger);
00390     _demand._itDTD = lDTD;
00391     //STDAIR_LOG_DEBUG ("DTD: " << lDTD);
00392 }
00393
00394 // //////////////////////////////////////
00395 storeDTDProbMass::storeDTDProbMass (DemandStruct& ioDemand)
00396     : ParserSemanticAction (ioDemand) {
00397 }
00398
00399 // //////////////////////////////////////
00400 void storeDTDProbMass::operator() (double iReal) const {
00401     const stdair::FloatDuration_T lZeroDTDFloat = 0.0;
00402     stdair::FloatDuration_T lDTDFloat =
00403         static_cast<stdair::FloatDuration_T> (_demand._itDTD);
00404     lDTDFloat = lZeroDTDFloat - lDTDFloat;
00405
00406     _demand._dtdProbDist.insert (ArrivalPatternCumulativeDistribution_T::
00407         value_type (lDTDFloat, iReal));
00408     //STDAIR_LOG_DEBUG ("DTDProbMass: " << iReal);
00409 }
00410
00411 // //////////////////////////////////////
00412 doEndDemand::doEndDemand (stdair::EventQueue& ioEventQueue,
00413     stdair::RandomGeneration& ioSharedGenerator,
00414     const POSProbabilityMass_T& iPOSProbMass,
00415     DemandStruct& ioDemand)
00416     : ParserSemanticAction (ioDemand), _eventQueue (ioEventQueue),
00417     _uniformGenerator (ioSharedGenerator),
00418     _posProbabilityMass (iPOSProbMass) {
00419 }
00420
00421 // //////////////////////////////////////
00422 // void doEndDemand::operator() (char iChar) const {
00423 void doEndDemand::operator() (iterator_t iStr, iterator_t iStrEnd) const {
00424
00425     // DEBUG: Display the result
00426     // STDAIR_LOG_DEBUG ("Demand: " << _demand.describe());
00427
00428     // Create the Demand BOM objects
00429     DemandManager::createDemandCharacteristics (_eventQueue, _uniformGenerator
00430         ,
00431         _posProbabilityMass, _demand)
00432
00433     // Clean the lists
00434     _demand._posProbDist.clear();
00435     _demand._channelProbDist.clear();
00436     _demand._tripProbDist.clear();
00437     _demand._stayProbDist.clear();
00438     _demand._ffProbDist.clear();
00439     _demand._prefDepTimeProbDist.clear();
00440     _demand._timeValueProbDist.clear();
00441     _demand._dtdProbDist.clear();
00442 }
00443
00444 // //////////////////////////////////////
00445 //
00446 // Utility Parsers
00447 //
00448 // //////////////////////////////////////
00449 intl_p_t intl_p;
00450
00451 uint2_p_t uint2_p;
00452
00453 uint1_2_p_t uint1_2_p;
00454
00455 uint1_3_p_t uint1_3_p;
00456
00457 uint4_p_t uint4_p;
00458
00459
00460
00461
00462
00463

```

```

00465     uint1_4_p_t uint1_4_p;
00466
00468     repeat_p_t airline_code_p (chset_t("0-9A-Z").derived(), 2, 3);
00469
00471     bounded1_4_p_t flight_number_p (uint1_4_p.derived(), 0u, 9999u);
00472
00474     bounded4_p_t year_p (uint4_p.derived(), 2000u, 2099u);
00475
00477     bounded2_p_t month_p (uint2_p.derived(), 1u, 12u);
00478
00480     bounded2_p_t day_p (uint2_p.derived(), 1u, 31u);
00481
00483     repeat_p_t dow_p (chset_t("0-1").derived().derived(), 7, 7);
00484
00486     repeat_p_t airport_p (chset_t("0-9A-Z").derived(), 3, 3);
00487
00489     bounded1_2_p_t hours_p (uint1_2_p.derived(), 0u, 23u);
00490
00492     bounded2_p_t minutes_p (uint2_p.derived(), 0u, 59u);
00493
00495     bounded2_p_t seconds_p (uint2_p.derived(), 0u, 59u);
00496
00498     chset_t cabin_code_p ("A-Z");
00499
00501     chset_t passenger_type_p ("A-Z");
00502
00504     chset_t ff_type_p ("A-Z");
00505
00507     int1_p_t family_code_p;
00508
00510     repeat_p_t class_code_list_p (chset_t("A-Z").derived(), 1, 26);
00511
00513     bounded1_3_p_t stay_duration_p (uint1_3_p.derived(), 0u, 999u);
00514
00515
00516     // //////////////////////////////////////
00517     // (Boost Spirit) Grammar Definition
00518     // //////////////////////////////////////
00519
00520     // //////////////////////////////////////
00521     DemandParser::DemandParser (stdair::EventQueue& ioEventQueue,
00522                                 stdair::RandomGeneration& ioSharedGenerator,
00523                                 const POSProbabilityMass_T& iPOSProbMass,
00524                                 DemandStruct& ioDemand)
00525     : _eventQueue (ioEventQueue), _uniformGenerator (ioSharedGenerator),
00526       _posProbabilityMass (iPOSProbMass), _demand (ioDemand) {
00527     }
00528
00529     // //////////////////////////////////////
00530     template<typename ScannerT>
00531     DemandParser::definition<ScannerT>::
00532     definition (DemandParser const& self) {
00533
00534         demand_list = *( not_to_be_parsed |
00535                        demand)
00536
00537         ;
00538
00539         not_to_be_parsed = bsc::
00540             lexeme_d[bsc::comment_p("//")
00541                   | bsc::comment_p("/*", "*/")
00542                   | bsc::eol_p]
00543
00544         ;
00545
00546         demand =
00547             pref_dep_date_range
00548             >> ';' >> origin >> ';' >> destination
00549             >> ';' >> pref_cabin[storePrefCabin(self._demand)]
00550             >> ';' >> pos_dist
00551             >> ';' >> channel_dist
00552             >> ';' >> trip_dist
00553             >> ';' >> stay_dist
00554             >> ';' >> ff_dist
00555             >> ';' >> pref_dep_time_dist
00556             >> ';' >> wtp
00557             >> ';' >> time_value_dist
00558             >> ';' >> dtd_dist
00559             >> ';' >> demand_params
00560             >> demand_end[doEndDemand (self._eventQueue, self._uniformGenerator,
00561                                     self._posProbabilityMass, self._demand)]
00562
00563         ;
00564
00565         demand_end = bsc::ch_p(';')
00566
00567         ;
00568
00569         pref_dep_date_range = date[storePrefDepDateRangeStart(self._demand)]
00570             >> ';' >> date[storePrefDepDateRangeEnd(self._demand)]
00571             >> ';' >> dow[storeDow(self._demand)]

```



```

00568         ;
00569
00570     date =
00571         bsc::lexeme_d[ (year_p) [bsc::assign_a(self._demand._itYear)]
00572         >> '-' >> (month_p) [bsc::assign_a(self._demand._itMonth)]
00573         >> '-' >> (day_p) [bsc::assign_a(self._demand._itDay)]
00574     ]
00575     ;
00576
00577     dow = bsc::lexeme_d[ dow_p ]
00578     ;
00579
00580     origin =
00581         (airport_p) [storeOrigin(self._demand)]
00582     ;
00583
00584     destination =
00585         (airport_p) [storeDestination(self._demand)]
00586     ;
00587
00588     pref_cabin = cabin_code_p;
00589
00590     pos_dist =
00591         pos_pair >> *( ',' >> pos_pair )
00592     ;
00593
00594     pos_pair =
00595         pos_code[storePosCode(self._demand)]
00596         >> ':' >> pos_share
00597     ;
00598
00599     pos_code =
00600         airport_p
00601         | bsc::chseq_p("row")
00602     ;
00603
00604     pos_share =
00605         (bsc::ureal_p) [storePosProbMass(self._demand)]
00606     ;
00607
00608     channel_dist =
00609         channel_pair >> *( ',' >> channel_pair )
00610     ;
00611
00612     channel_pair =
00613         channel_code[storeChannelCode(self._demand)]
00614         >> ':' >> channel_share
00615     ;
00616
00617     channel_code =
00618         bsc::chseq_p("DF") | bsc::chseq_p("DN")
00619         | bsc::chseq_p("IF") | bsc::chseq_p("IN")
00620     ;
00621
00622     channel_share =
00623         (bsc::ureal_p) [storeChannelProbMass(self._demand)]
00624     ;
00625
00626     trip_dist =
00627         trip_pair >> *( ',' >> trip_pair )
00628     ;
00629
00630     trip_pair =
00631         trip_code[storeTripCode(self._demand)]
00632         >> ':' >> trip_share
00633     ;
00634
00635     trip_code =
00636         bsc::chseq_p("RO") | bsc::chseq_p("RI") | bsc::chseq_p("OW")
00637     ;
00638
00639     trip_share =
00640         (bsc::ureal_p) [storeTripProbMass(self._demand)]
00641     ;
00642
00643     stay_dist =
00644         stay_pair >> *( ',' >> stay_pair )
00645     ;
00646
00647     stay_pair =
00648         (stay_duration_p) [storeStayCode(self._demand)]
00649         >> ':' >> stay_share
00650     ;
00651
00652     stay_share =
00653         (bsc::ureal_p) [storeStayProbMass(self._demand)]
00654     ;

```

```

00655
00656     ff_dist =
00657         ff_pair >> *( ' ,' >> ff_pair )
00658     ;
00659
00660     ff_pair =
00661         ff_code[storeFFCode(self._demand)]
00662         >> ' :' >> ff_share
00663     ;
00664
00665     ff_code = ff_type_p;
00666
00667     ff_share =
00668         (bsc::ureal_p)[storeFFProbMass(self._demand)]
00669     ;
00670
00671     pref_dep_time_dist =
00672         pref_dep_time_pair >> *( ' ,' >> pref_dep_time_pair )
00673     ;
00674
00675     pref_dep_time_pair =
00676         (time)[storePrefDepTime(self._demand)]
00677         >> ' :' >> pref_dep_time_share
00678     ;
00679
00680     pref_dep_time_share =
00681         (bsc::ureal_p)[storePrefDepTimeProbMass(self._demand)]
00682     ;
00683
00684     time =
00685         bsc::lexeme_d[
00686             (hours_p)[bsc::assign_a(self._demand._itHours)]
00687             >> !(' .' >> (minutes_p)[bsc::assign_a(self._demand._itMinutes)])
00688             >> !(' .' >> (seconds_p)[bsc::assign_a(self._demand._itSeconds)])
00689         ]
00690     ;
00691
00692     wtp =
00693         (bsc::ureal_p)[storeWTP(self._demand)]
00694     ;
00695
00696     time_value_dist =
00697         time_value_pair >> *( ' ,' >> time_value_pair )
00698     ;
00699
00700     time_value_pair =
00701         (bsc::ureal_p)[storeTimeValue(self._demand)]
00702         >> ' :' >> time_value_share
00703     ;
00704
00705     time_value_share =
00706         (bsc::ureal_p)[storeTimeValueProbMass(self._demand)]
00707     ;
00708
00709     dtd_dist =
00710         dtd_pair >> *( ' ,' >> dtd_pair )
00711     ;
00712
00713     dtd_pair =
00714         (bsc::ureal_p)[storeDTD(self._demand)]
00715         >> ' :' >> dtd_share
00716     ;
00717
00718     dtd_share =
00719         (bsc::ureal_p)[storeDTDProbMass(self._demand)]
00720     ;
00721
00722     demand_params =
00723         bsc::ch_p('N')
00724         >> ' ,'
00725         >> (bsc::ureal_p)[storeDemandMean(self._demand)]
00726         >> ' ,'
00727         >> (bsc::ureal_p)[storeDemandStdDev(self._demand)]
00728     ;
00729
00730     // BOOST_SPIRIT_DEBUG_NODE (DemandParser);
00731     BOOST_SPIRIT_DEBUG_NODE (demand_list);
00732     BOOST_SPIRIT_DEBUG_NODE (not_to_be_parsed);
00733     BOOST_SPIRIT_DEBUG_NODE (demand);
00734     BOOST_SPIRIT_DEBUG_NODE (demand_end);
00735     BOOST_SPIRIT_DEBUG_NODE (pref_dep_date);
00736     BOOST_SPIRIT_DEBUG_NODE (date);
00737     BOOST_SPIRIT_DEBUG_NODE (origin);
00738     BOOST_SPIRIT_DEBUG_NODE (destination);
00739     BOOST_SPIRIT_DEBUG_NODE (pref_cabin);
00740     BOOST_SPIRIT_DEBUG_NODE (pos_dist);
00741     BOOST_SPIRIT_DEBUG_NODE (pos_pair);

```

```

00742     BOOST_SPIRIT_DEBUG_NODE (pos_code);
00743     BOOST_SPIRIT_DEBUG_NODE (pos_share);
00744     BOOST_SPIRIT_DEBUG_NODE (channel_dist);
00745     BOOST_SPIRIT_DEBUG_NODE (channel_pair);
00746     BOOST_SPIRIT_DEBUG_NODE (channel_code);
00747     BOOST_SPIRIT_DEBUG_NODE (channel_share);
00748     BOOST_SPIRIT_DEBUG_NODE (trip_dist);
00749     BOOST_SPIRIT_DEBUG_NODE (trip_pair);
00750     BOOST_SPIRIT_DEBUG_NODE (trip_code);
00751     BOOST_SPIRIT_DEBUG_NODE (trip_share);
00752     BOOST_SPIRIT_DEBUG_NODE (stay_dist);
00753     BOOST_SPIRIT_DEBUG_NODE (stay_pair);
00754     BOOST_SPIRIT_DEBUG_NODE (stay_share);
00755     BOOST_SPIRIT_DEBUG_NODE (ff_dist);
00756     BOOST_SPIRIT_DEBUG_NODE (ff_pair);
00757     BOOST_SPIRIT_DEBUG_NODE (ff_code);
00758     BOOST_SPIRIT_DEBUG_NODE (ff_share);
00759     BOOST_SPIRIT_DEBUG_NODE (pref_dep_time_dist);
00760     BOOST_SPIRIT_DEBUG_NODE (pref_dep_time_pair);
00761     BOOST_SPIRIT_DEBUG_NODE (pref_dep_time_share);
00762     BOOST_SPIRIT_DEBUG_NODE (time);
00763     BOOST_SPIRIT_DEBUG_NODE (wtp);
00764     BOOST_SPIRIT_DEBUG_NODE (time_value_dist);
00765     BOOST_SPIRIT_DEBUG_NODE (time_value_pair);
00766     BOOST_SPIRIT_DEBUG_NODE (time_value_share);
00767     BOOST_SPIRIT_DEBUG_NODE (dtd_dist);
00768     BOOST_SPIRIT_DEBUG_NODE (dtd_pair);
00769     BOOST_SPIRIT_DEBUG_NODE (dtd_share);
00770     BOOST_SPIRIT_DEBUG_NODE (demand_params);
00771 }
00772
00773 // //////////////////////////////////////
00774 template<typename ScannerT>
00775 bsc::rule<ScannerT> const&
00776 DemandParser::definition<ScannerT>::start() const {
00777     return demand_list;
00778 }
00779
00780 }
00781
00782 //
00783 // Entry class for the file parser
00784 //
00785 // //////////////////////////////////////
00786 DemandFileParser::
00787 DemandFileParser (stdair::EventQueue& ioEventQueue,
00788                  stdair::RandomGeneration& ioSharedGenerator,
00789                  const POSProbabilityMass_T& iPOSProbMass,
00790                  const std::string& iFilename)
00791 : _filename (iFilename), _eventQueue (ioEventQueue),
00792   _uniformGenerator (ioSharedGenerator),
00793   _posProbabilityMass (iPOSProbMass) {
00794     init();
00795 }
00796
00797 // //////////////////////////////////////
00798 void DemandFileParser::init() {
00799     // Check that the file exists and is readable
00800     const bool doesExistAndIsReadable =
00801         stdair::BasFileMgr::doesExistAndIsReadable (_filename);
00802
00803     if (doesExistAndIsReadable == false) {
00804         STDAIR_LOG_ERROR ("The demand file " << _filename
00805             << " does not exist or can not be read.");
00806
00807         throw DemandInputFileNotFoundException ("The demand file " + _filename
00808             + " does not exist or can not "
00809             + "be read");
00810     }
00811
00812     // Open the file
00813     _startIterator = iterator_t (_filename);
00814
00815     // Check the filename exists and can be open
00816     if (!_startIterator) {
00817         STDAIR_LOG_ERROR ("The demand file " << _filename << " can not be open.");
00818     }
00819
00820     throw DemandInputFileNotFoundException ("The demand file " + _filename
00821         + " does not exist or can not "
00822         + "be read");
00823
00824     // Create an EOF iterator
00825     _endIterator = _startIterator.make_end();

```

```

00830     }
00831
00832     // //////////////////////////////////////
00833     bool DemandFileParser::generateDemand () {
00834         bool oResult = false;
00835
00836         STDAIR_LOG_DEBUG ("Parsing demand input file: " << _filename);
00837
00838         // Initialise the parser (grammar) with the helper/staging structure.
00839         DemandParserHelper::DemandParser lDemandParser (_eventQueue,
00840                                                         _uniformGenerator,
00841                                                         _posProbabilityMass,
00842                                                         _demand);
00843
00844         // Launch the parsing of the file and, thanks to the doEndDemand
00845         // call-back structure, the building of the whole EventQueue BOM
00846         // (i.e., including Inventory, FlightDate, LegDate, SegmentDate, etc.)
00847         bsc::parse_info<iterator_t> info =
00848             bsc::parse (_startIterator, _endIterator, lDemandParser,
00849                       bsc::space_p - bsc::eol_p);
00850
00851         // Retrieves whether or not the parsing was successful
00852         oResult = info.hit;
00853
00854         const std::string hasBeenFullyReadStr = (info.full == true)?"":"not ";
00855         if (oResult == true) {
00856             STDAIR_LOG_DEBUG ("Parsing of demand input file: " << _filename
00857                               << " succeeded: read " << info.length
00858                               << " characters. The input file has "
00859                               << hasBeenFullyReadStr
00860                               << "been fully read. Stop point: " << info.stop);
00861         } else {
00862             std::ostringstream oStr;
00863             oStr << "Parsing of demand input file: " << _filename << " failed: read "
00864                 << info.length << " characters. The input file has "
00865                 << hasBeenFullyReadStr << "been fully read. Stop point: "
00866                 << info.stop;
00867             STDAIR_LOG_ERROR (oStr.str());
00868             throw stdair::ParserException (oStr.str());
00869         }
00870     }
00871
00872     return oResult;
00873 }
00874
00875 }

```

25.93 trademgen/command/DemandParserHelper.hpp File Reference

```

#include <string>
#include <stdair/command/CmdAbstract.hpp>
#include <trademgen/TRADEMGENTypes.hpp>
#include <trademgen/basic/BasParserTypes.hpp>
#include <trademgen/bom/DemandStruct.hpp>

```

Classes

- struct [TRADEMGENT::DemandParserHelper::ParserSemanticAction](#)
- struct [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeStart](#)
- struct [TRADEMGENT::DemandParserHelper::storePrefDepDateRangeEnd](#)
- struct [TRADEMGENT::DemandParserHelper::storeDow](#)
- struct [TRADEMGENT::DemandParserHelper::storeOrigin](#)
- struct [TRADEMGENT::DemandParserHelper::storeDestination](#)
- struct [TRADEMGENT::DemandParserHelper::storePrefCabin](#)
- struct [TRADEMGENT::DemandParserHelper::storeDemandMean](#)
- struct [TRADEMGENT::DemandParserHelper::storeDemandStdDev](#)
- struct [TRADEMGENT::DemandParserHelper::storePosCode](#)
- struct [TRADEMGENT::DemandParserHelper::storePosProbMass](#)
- struct [TRADEMGENT::DemandParserHelper::storeChannelCode](#)
- struct [TRADEMGENT::DemandParserHelper::storeChannelProbMass](#)

- struct TRADEMGEN::DemandParserHelper::storeTripCode
- struct TRADEMGEN::DemandParserHelper::storeTripProbMass
- struct TRADEMGEN::DemandParserHelper::storeStayCode
- struct TRADEMGEN::DemandParserHelper::storeStayProbMass
- struct TRADEMGEN::DemandParserHelper::storeFFCode
- struct TRADEMGEN::DemandParserHelper::storeFFProbMass
- struct TRADEMGEN::DemandParserHelper::storePrefDepTime
- struct TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass
- struct TRADEMGEN::DemandParserHelper::storeWTP
- struct TRADEMGEN::DemandParserHelper::storeTimeValue
- struct TRADEMGEN::DemandParserHelper::storeTimeValueProbMass
- struct TRADEMGEN::DemandParserHelper::storeDTD
- struct TRADEMGEN::DemandParserHelper::storeDTDProbMass
- struct TRADEMGEN::DemandParserHelper::doEndDemand
- struct TRADEMGEN::DemandParserHelper::DemandParser
- struct TRADEMGEN::DemandParserHelper::DemandParser::definition< ScannerT >
- class TRADEMGEN::DemandFileParser

Namespaces

- namespace stdair
 Forward declarations.
- namespace TRADEMGEN
- namespace TRADEMGEN::DemandParserHelper

25.94 DemandParserHelper.hpp

```

00001 #ifndef __TRADEMGEN_CMD_DEMANDPARSERHELPER_HPP
00002 #define __TRADEMGEN_CMD_DEMANDPARSERHELPER_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // STDAIR
00010 #include <stdair/command/CmdAbstract.hpp>
00011 // TRADEMGEN
00012 #include <trademgen/TRADEMGEN_Types.hpp>
00013 #include <trademgen/basic/BasParserTypes.hpp>
00014 #include <trademgen/bom/DemandStruct.hpp>
00015
00016 // Forward declarations
00017 namespace stdair {
00018     class EventQueue;
00019     struct RandomGeneration;
00020 }
00021
00022 namespace TRADEMGEN {
00023
00024     namespace DemandParserHelper {
00025
00026         // //////////////////////////////////////
00027         // Semantic actions
00028         // //////////////////////////////////////
00030         struct ParserSemanticAction {
00032             ParserSemanticAction (DemandStruct&);
00034             DemandStruct& _demand;
00035         };
00036
00038         struct storePrefDepDateRangeStart : public ParserSemanticAction {
00040             storePrefDepDateRangeStart (DemandStruct&);
00042             void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00043         };
00044
00046         struct storePrefDepDateRangeEnd : public ParserSemanticAction {
00048             storePrefDepDateRangeEnd (DemandStruct&);
00050             void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00051         };

```

```

00052
00054     struct storeDow : public ParserSemanticAction {
00056         storeDow (DemandStruct&);
00058         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00059     };
00060
00062     struct storeOrigin : public ParserSemanticAction {
00064         storeOrigin (DemandStruct&);
00066         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00067     };
00068
00070     struct storeDestination : public ParserSemanticAction {
00072         storeDestination (DemandStruct&);
00074         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00075     };
00076
00078     struct storePrefCabin : public ParserSemanticAction {
00080         storePrefCabin (DemandStruct&);
00082         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00083     };
00084
00086     struct storeDemandMean : public ParserSemanticAction {
00088         storeDemandMean (DemandStruct&);
00090         void operator() (double iReal) const;
00091     };
00092
00094     struct storeDemandStdDev : public ParserSemanticAction {
00096         storeDemandStdDev (DemandStruct&);
00098         void operator() (double iReal) const;
00099     };
00100
00102     struct storePosCode : public ParserSemanticAction {
00104         storePosCode (DemandStruct&);
00106         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00107     };
00108
00110     struct storePosProbMass : public ParserSemanticAction {
00112         storePosProbMass (DemandStruct&);
00114         void operator() (double iReal) const;
00115     };
00116
00118     struct storeChannelCode : public ParserSemanticAction {
00120         storeChannelCode (DemandStruct&);
00122         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00123     };
00124
00126     struct storeChannelProbMass : public ParserSemanticAction {
00128         storeChannelProbMass (DemandStruct&);
00130         void operator() (double iReal) const;
00131     };
00132
00134     struct storeTripCode : public ParserSemanticAction {
00136         storeTripCode (DemandStruct&);
00138         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00139     };
00140
00142     struct storeTripProbMass : public ParserSemanticAction {
00144         storeTripProbMass (DemandStruct&);
00146         void operator() (double iReal) const;
00147     };
00148
00150     struct storeStayCode : public ParserSemanticAction {
00152         storeStayCode (DemandStruct&);
00154         void operator() (unsigned int iInteger) const;
00155     };
00156
00158     struct storeStayProbMass : public ParserSemanticAction {
00160         storeStayProbMass (DemandStruct&);
00162         void operator() (double iReal) const;
00163     };
00164
00166     struct storeFFCode : public ParserSemanticAction {
00168         storeFFCode (DemandStruct&);
00170         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00171     };
00172
00174     struct storeFFProbMass : public ParserSemanticAction {
00176         storeFFProbMass (DemandStruct&);
00178         void operator() (double iReal) const;
00179     };
00180
00183     struct storePrefDepTime : public ParserSemanticAction {
00185         storePrefDepTime (DemandStruct&);
00187         void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00188     };
00189
00192     struct storePrefDepTimeProbMass : public ParserSemanticAction {

```

```

00194     storePrefDepTimeProbMass (DemandStruct&);
00196     void operator() (double iReal) const;
00197 };
00198
00200 struct storeWTP : public ParserSemanticAction {
00202     storeWTP (DemandStruct&);
00204     void operator() (double iReal) const;
00205 };
00206
00208 struct storeTimeValue : public ParserSemanticAction {
00210     storeTimeValue (DemandStruct&);
00212     void operator() (double iReal) const;
00213 };
00214
00216 struct storeTimeValueProbMass : public ParserSemanticAction {
00218     storeTimeValueProbMass (DemandStruct&);
00220     void operator() (double iReal) const;
00221 };
00222
00225 struct storeDTD : public ParserSemanticAction {
00227     storeDTD (DemandStruct&);
00229     void operator() (unsigned int iInteger) const;
00230 };
00231
00234 struct storeDTDProbMass : public ParserSemanticAction {
00236     storeDTDProbMass (DemandStruct&);
00238     void operator() (double iReal) const;
00239 };
00240
00242 struct doEndDemand : public ParserSemanticAction {
00244     doEndDemand (stdair::EventQueue&, stdair::RandomGeneration&,
00245                 const POSProbabilityMass_T&, DemandStruct&);
00247     void operator() (iterator_t iStr, iterator_t iStrEnd) const;
00249     stdair::EventQueue& _eventQueue;
00250     stdair::RandomGeneration& _uniformGenerator;
00251     const POSProbabilityMass_T& _posProbabilityMass;
00252 };
00253
00254
00256 //
00257 // (Boost Spirit) Grammar Definition
00258 //
00260
00345 struct DemandParser :
00346     public boost::spirit::classic::grammar<DemandParser> {
00347
00348     DemandParser (stdair::EventQueue&, stdair::RandomGeneration&,
00349                 const POSProbabilityMass_T&, DemandStruct&);
00350
00351     template <typename ScannerT>
00352     struct definition {
00353         definition (DemandParser const& self);
00354
00355         // Instantiation of rules
00356         boost::spirit::classic::rule<ScannerT> demand_list,
00357             not_to_be_parsed, demand, demand_end, pref_dep_date_range,
00358             date, dow, origin, destination, pref_cabin, demand_params,
00359             pos_dist, pos_pair, pos_code, pos_share,
00360             channel_dist, channel_pair, channel_code, channel_share,
00361             trip_dist, trip_pair, trip_code, trip_share,
00362             stay_dist, stay_pair, stay_share,
00363             ff_dist, ff_pair, ff_code, ff_share,
00364             pref_dep_time_dist, pref_dep_time_pair, pref_dep_time_share, time,
00365             wtp,
00366             time_value_dist, time_value_pair, time_value_share,
00367             dtd_dist, dtd_pair, dtd_share;
00368
00370         boost::spirit::classic::rule<ScannerT> const& start() const;
00371     };
00372
00373     // Parser Context
00374     stdair::EventQueue& _eventQueue;
00375     stdair::RandomGeneration& _uniformGenerator;
00376     const POSProbabilityMass_T& _posProbabilityMass;
00377     DemandStruct& _demand;
00378 };
00379
00380 }
00381
00382
00384 //
00385 // Entry class for the file parser
00386 //
00388
00393 class DemandFileParser : public stdair::CmdAbstract {
00394 public:
00396     DemandFileParser (stdair::EventQueue&, stdair::RandomGeneration&,

```

```

00397         const POSProbabilityMass_T&,
00398         const stdair::Filename_T& iDemandInputFilename);
00399
00401     bool generateDemand ();
00402
00403 private:
00404     void init ();
00405
00406 private:
00407     // Attributes
00410     stdair::Filename_T _filename;
00411
00413     iterator_t _startIterator;
00414
00416     iterator_t _endIterator;
00417
00419     stdair::EventQueue& _eventQueue;
00420
00422     stdair::RandomGeneration& _uniformGenerator;
00423
00425     const POSProbabilityMass_T& _posProbabilityMass;
00426
00428     DemandStruct _demand;
00429 };
00430
00431 }
00432 #endif // __TRADEMGEN_CMD_DEMANDPARSERHELPER_HPP

```

25.95 trademgen/config/trademgen-paths.hpp File Reference

Defines

- #define [PACKAGE](#) "trademgen"
- #define [PACKAGE_NAME](#) "TRADEMGEN"
- #define [PACKAGE_VERSION](#) "0.2.2"
- #define [PREFIXDIR](#) "/usr"
- #define [EXEC_PREFIX](#) "/usr"
- #define [BINDIR](#) "/usr/bin"
- #define [LIBDIR](#) "/usr/lib"
- #define [LIBEXECDIR](#) "/usr/libexec"
- #define [SBINDIR](#) "/usr/sbin"
- #define [SYSCONFDIR](#) "/usr/etc"
- #define [INCLUDEDIR](#) "/usr/include"
- #define [DATAROOTDIR](#) "/usr/share"
- #define [DATADIR](#) "/usr/share"
- #define [DOCDIR](#) "/usr/share/doc/trademgen-0.2.2"
- #define [MANDIR](#) "/usr/share/man"
- #define [INFODIR](#) "/usr/share/info"
- #define [HTMLDIR](#) "/usr/share/doc/trademgen-0.2.2/html"
- #define [PDFDIR](#) "/usr/share/doc/trademgen-0.2.2/html"
- #define [STDAIR_SAMPLE_DIR](#) "/usr/share/stdair/samples"

25.95.1 Define Documentation

25.95.1.1 #define [PACKAGE](#) "trademgen"

Definition at line 4 of file [trademgen-paths.hpp](#).

25.95.1.2 #define [PACKAGE_NAME](#) "TRADEMGEN"

Definition at line 5 of file [trademgen-paths.hpp](#).

Referenced by [readConfiguration\(\)](#).

25.95.1.3 `#define PACKAGE_VERSION "0.2.2"`

Definition at line 6 of file [trademgen-paths.hpp](#).

Referenced by [readConfiguration\(\)](#).

25.95.1.4 `#define PREFIXDIR "/usr"`

Definition at line 7 of file [trademgen-paths.hpp](#).

Referenced by [readConfiguration\(\)](#).

25.95.1.5 `#define EXEC_PREFIX "/usr"`

Definition at line 8 of file [trademgen-paths.hpp](#).

25.95.1.6 `#define BINDIR "/usr/bin"`

Definition at line 9 of file [trademgen-paths.hpp](#).

25.95.1.7 `#define LIBDIR "/usr/lib"`

Definition at line 10 of file [trademgen-paths.hpp](#).

25.95.1.8 `#define LIBEXECDIR "/usr/libexec"`

Definition at line 11 of file [trademgen-paths.hpp](#).

25.95.1.9 `#define SBINDIR "/usr/sbin"`

Definition at line 12 of file [trademgen-paths.hpp](#).

25.95.1.10 `#define SYSCONFDIR "/usr/etc"`

Definition at line 13 of file [trademgen-paths.hpp](#).

25.95.1.11 `#define INCLUDEDIR "/usr/include"`

Definition at line 14 of file [trademgen-paths.hpp](#).

25.95.1.12 `#define DATAROOTDIR "/usr/share"`

Definition at line 15 of file [trademgen-paths.hpp](#).

25.95.1.13 `#define DATADIR "/usr/share"`

Definition at line 16 of file [trademgen-paths.hpp](#).

25.95.1.14 `#define DOCDIR "/usr/share/doc/trademgen-0.2.2"`

Definition at line 17 of file [trademgen-paths.hpp](#).

25.95.1.15 `#define MANDIR "/usr/share/man"`

Definition at line 18 of file [trademgen-paths.hpp](#).

25.95.1.16 `#define INFODIR "/usr/share/info"`

Definition at line 19 of file [trademgen-paths.hpp](#).

25.95.1.17 `#define HTMLDIR "/usr/share/doc/trademgen-0.2.2/html"`

Definition at line 20 of file [trademgen-paths.hpp](#).

25.95.1.18 `#define PDFDIR "/usr/share/doc/trademgen-0.2.2/html"`

Definition at line 21 of file [trademgen-paths.hpp](#).

25.95.1.19 `#define STDAIR_SAMPLE_DIR "/usr/share/stdair/samples"`

Definition at line 22 of file [trademgen-paths.hpp](#).

25.96 trademgen-paths.hpp

```
00001 #ifndef __TRADEMGEN_PATHS_HPP__
00002 #define __TRADEMGEN_PATHS_HPP__
00003
00004 #define PACKAGE "trademgen"
00005 #define PACKAGE_NAME "TRADEMGEN"
00006 #define PACKAGE_VERSION "0.2.2"
00007 #define PREFIXDIR "/usr"
00008 #define EXEC_PREFIX "/usr"
00009 #define BINDIR "/usr/bin"
00010 #define LIBDIR "/usr/lib"
00011 #define LIBEXECDIR "/usr/libexec"
00012 #define SBINDIR "/usr/sbin"
00013 #define SYSCONFDIR "/usr/etc"
00014 #define INCLUDEDIR "/usr/include"
00015 #define DATAROOTDIR "/usr/share"
00016 #define DATADIR "/usr/share"
00017 #define DOCDIR "/usr/share/doc/trademgen-0.2.2"
00018 #define MANDIR "/usr/share/man"
00019 #define INFODIR "/usr/share/info"
00020 #define HTMLDIR "/usr/share/doc/trademgen-0.2.2/html"
00021 #define PDFDIR "/usr/share/doc/trademgen-0.2.2/html"
00022 #define STDAIR_SAMPLE_DIR "/usr/share/stdair/samples"
00023
00024 #endif // __TRADEMGEN_PATHS_HPP__
```

25.97 trademgen/DBParams.hpp File Reference

```
#include <iosfwd>
#include <string>
#include <trademgen/TRADEMGEN_Types.hpp>
#include <trademgen/TRADEMGEN_Abstract.hpp>
```

Classes

- struct [TRADEMGEN::DBParams](#)

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef std::list< std::string > [TRADEMGEN::DBParamsNameList_T](#)

25.98 DBParams.hpp

```
00001 #ifndef __TRADEMGEN_DBPARAMS_HPP
00002 #define __TRADEMGEN_DBPARAMS_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
```

```

00008 #include <iosfwd>
00009 #include <string>
00010 // Trademgen
00011 #include <trademgen/TRADEMGEN_Types.hpp>
00012 #include <trademgen/TRADEMGEN_Abstract.hpp>
00013
00014 namespace TRADEMGEN {
00015
00017     typedef std::list<std::string> DBParamsNameList_T;
00018
00019
00021     struct DBParams : public TRADEMGEN_Abstract {
00022     public:
00023         // /////////// Getters ///////////
00025         std::string getUser() const {
00026             return _user;
00027         }
00028
00030         std::string getPassword() const {
00031             return _passwd;
00032         }
00033
00035         std::string getHost() const {
00036             return _host;
00037         }
00038
00040         std::string getPort() const {
00041             return _port;
00042         }
00043
00045         std::string getDBName() const {
00046             return _dbname;
00047         }
00048
00049
00050         // /////////// Setters ///////////
00052         void setUser (const std::string& iUser) {
00053             _user = iUser;
00054         }
00055
00057         void setPassword (const std::string& iPasswd) {
00058             _passwd = iPasswd;
00059         }
00060
00062         void setHost (const std::string& iHost) {
00063             _host = iHost;
00064         }
00065
00067         void setPort (const std::string& iPort) {
00068             _port = iPort;
00069         }
00070
00072         void setDBName (const std::string& iDBName) {
00073             _dbname = iDBName;
00074         }
00075
00076
00077     public:
00078         // /////////// Busines methods ///////////
00080         bool check () const {
00081             if (_user.empty() == true || _passwd.empty() == true
00082                 || _host.empty() == true || _port.empty()
00083                 || _dbname.empty() == true) {
00084                 return false;
00085             }
00086             return true;
00087         }
00088
00089     public:
00090         // /////////// Display methods ///////////
00093         void toStream (std::ostream& ioOut) const {
00094             ioOut << toString();
00095         }
00096
00099         void fromStream (std::istream&) {
00100         }
00101
00103         std::string toShortString() const {
00104             std::ostringstream oStr;
00105             oStr << _dbname << "." << _user << "@" << _host << ":" << _port;
00106             return oStr.str();
00107         }
00108
00110         std::string toString() const {
00111             std::ostringstream oStr;
00112             oStr << _dbname << "." << _user << "@" << _host << ":" << _port;
00113             return oStr.str();

```

```

00114     }
00115
00116
00117 public:
00118     DBParams (const std::string& iDBUser, const std::string& iDBPasswd,
00119               const std::string& iDBHost, const std::string& iDBPort,
00120               const std::string& iDBName)
00121     : _user (iDBUser), _passwd (iDBPasswd), _host (iDBHost), _port (iDBPort),
00122       _dbname (iDBName) {
00123     }
00124
00125
00126     // DBParams ();
00127     // DBParams (const DBParams&);
00128
00129     virtual ~DBParams() {}
00130
00131 private:
00132     // ////////// Attributes //////////
00133     std::string _user;
00134     std::string _passwd;
00135     std::string _host;
00136     std::string _port;
00137     std::string _dbname;
00138 };
00139 }
00140 #endif // __TRADEMGEN_DBPARAMS_HPP

```

25.99 trademgen/factory/FacTRADEMGENSEerviceContext.cpp File Reference

```

#include <cassert>
#include <stdair/service/FacSupervisor.hpp>
#include <trademgen/factory/FacTRADEMGENSEerviceContext.hpp>
#include <trademgen/service/TRADEMGEN_ServiceContext.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.100 FacTRADEMGENSEerviceContext.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // StdAir
00007 #include <stdair/service/FacSupervisor.hpp>
00008 // TraDemGen
00009 #include <trademgen/factory/FacTRADEMGENSEerviceContext.hpp>
00010 #include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
00011
00012 namespace TRADEMGEN {
00013
00014     FacTRADEMGENSEerviceContext* FacTRADEMGENSEerviceContext::_instance = NULL;
00015
00016     // //////////////////////////////////////
00017     FacTRADEMGENSEerviceContext::~FacTRADEMGENSEerviceContext() {
00018         _instance = NULL;
00019     }
00020
00021     // //////////////////////////////////////
00022     FacTRADEMGENSEerviceContext& FacTRADEMGENSEerviceContext::instance() {
00023
00024         if (_instance == NULL) {
00025             _instance = new FacTRADEMGENSEerviceContext();
00026             assert (_instance != NULL);
00027
00028             stdair::FacSupervisor::instance().registerServiceFactory (_instance);
00029         }
00030         return *_instance;
00031     }
00032
00033     // //////////////////////////////////////

```

```

00034 TRADEMGEN_ServiceContext& FacTRADEMGENSEerviceContext::
00035 create (const stdair::RandomSeed_T& iRandomSeed) {
00036     TRADEMGEN_ServiceContext* aServiceContext_ptr = NULL;
00037
00038     aServiceContext_ptr = new TRADEMGEN_ServiceContext (iRandomSeed);
00039     assert (aServiceContext_ptr != NULL);
00040
00041     // The new object is added to the Bom pool
00042     _pool.push_back (aServiceContext_ptr);
00043
00044     return *aServiceContext_ptr;
00045 }
00046
00047 }

```

25.101 trademgen/factory/FacTRADEMGENSEerviceContext.hpp File Reference

```

#include <stdair/stdair_maths_types.hpp>
#include <stdair/service/FacServiceAbstract.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>

```

Classes

- class [TRADEMGEN::FacTRADEMGENSEerviceContext](#)
Factory for creating the TraDemGen service context instance.

Namespaces

- namespace [TRADEMGEN](#)

25.102 FacTRADEMGENSEerviceContext.hpp

```

00001 #ifndef __TRADEMGEN_FAC_FACTRADEMGENSEERVICECONTEXT_HPP
00002 #define __TRADEMGEN_FAC_FACTRADEMGENSEERVICECONTEXT_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_maths_types.hpp>
00009 #include <stdair/service/FacServiceAbstract.hpp>
00010 // TraDemGen
00011 #include <trademgen/TRADEMGEN_Types.hpp>
00012
00013 namespace TRADEMGEN {
00014
00016     class TRADEMGEN_ServiceContext;
00017
00021     class FacTRADEMGENSEerviceContext : public stdair::FacServiceAbstract {
00022     public:
00030         static FacTRADEMGENSEerviceContext& instance();
00031
00038         ~FacTRADEMGENSEerviceContext();
00039
00048         TRADEMGEN_ServiceContext& create (const stdair::RandomSeed_T&);
00049
00051     protected:
00057         FacTRADEMGENSEerviceContext () {}
00058
00059     private:
00063         static FacTRADEMGENSEerviceContext* _instance;
00064     };
00065
00066 }
00067 #endif // __TRADEMGEN_FAC_FACTRADEMGENSEERVICECONTEXT_HPP

```

25.103 trademgen/python/pytrademgen.cpp File Reference

```
#include <cassert>
#include <stdexcept>
#include <fstream>
#include <sstream>
#include <string>
#include <list>
#include <vector>
#include <boost/python.hpp>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_exceptions.hpp>
#include <stdair/basic/BasFileMgr.hpp>
#include <stdair/basic/BasLogParams.hpp>
#include <stdair/basic/BasDBParams.hpp>
#include <trademgen/TRADEMGEN_Service.hpp>
```

Classes

- struct [TRADEMGEN::Trademgener](#)

Namespaces

- namespace [TRADEMGEN](#)

Functions

- [BOOST_PYTHON_MODULE](#) (libpytrademgen)

25.103.1 Function Documentation

25.103.1.1 [BOOST_PYTHON_MODULE](#) (libpytrademgen)

Definition at line 162 of file [pytrademgen.cpp](#).

References [TRADEMGEN::Trademgener::init\(\)](#), and [TRADEMGEN::Trademgener::trademgen\(\)](#).

25.104 pytrademgen.cpp

```
00001 // STL
00002 #include <cassert>
00003 #include <stdexcept>
00004 #include <fstream>
00005 #include <sstream>
00006 #include <string>
00007 #include <list>
00008 #include <vector>
00009 // Boost String
00010 #include <boost/python.hpp>
00011 // StdAir
00012 #include <stdair/stdair_basic_types.hpp>
00013 #include <stdair/stdair_exceptions.hpp>
00014 #include <stdair/basic/BasFileMgr.hpp>
00015 #include <stdair/basic/BasLogParams.hpp>
00016 #include <stdair/basic/BasDBParams.hpp>
00017 // TraDemGen
00018 #include <trademgen/TRADEMGEN_Service.hpp>
00019
00020 namespace TRADEMGEN {
00021
00022     struct Trademgener {
00023     public:
```

```

00025     std::string trademgen (const std::string& iQuery) {
00026         std::ostringstream oStream;
00027
00028         // Sanity check
00029         if (_logOutputStream == NULL) {
00030             oStream << "The log filepath is not valid." << std::endl;
00031             return oStream.str();
00032         }
00033         assert (_logOutputStream != NULL);
00034
00035         try {
00036
00037             // DEBUG
00038             *_logOutputStream << "Python search for '" << iQuery << "'
00039                 << std::endl;
00040
00041             if (_trademgenService == NULL) {
00042                 oStream << "The Trademgen service has not been initialised, "
00043                     << "i.e., the init() method has not been called "
00044                     << "correctly on the Trademgener object. Please "
00045                     << "check that all the parameters are not empty and "
00046                     << "point to actual files.";
00047                 *_logOutputStream << oStream.str();
00048                 return oStream.str();
00049             }
00050             assert (_trademgenService != NULL);
00051
00052             // Do the trademgen
00053             _trademgenService->displayAirlineListFromDB();
00054
00055             // DEBUG
00056             *_logOutputStream << "Python search for '" << iQuery
00057                 << "' returned '" << std::endl;
00058
00059             // DEBUG
00060             *_logOutputStream << "TraDemGen output: "
00061                 << oStream.str() << std::endl;
00062
00063         } catch (const stdair::RootException& eTrademgenError) {
00064             *_logOutputStream << "TraDemGen error: " << eTrademgenError.what()
00065                 << std::endl;
00066
00067         } catch (const std::exception& eStdError) {
00068             *_logOutputStream << "Error: " << eStdError.what() << std::endl;
00069
00070         } catch (...) {
00071             *_logOutputStream << "Unknown error" << std::endl;
00072         }
00073
00074         return oStream.str();
00075     }
00076
00077 public:
00079     Trademgener() : _trademgenService (NULL), _logOutputStream (NULL) {
00080     }
00081
00083     Trademgener (const Trademgener& iTrademgener)
00084         : _trademgenService (iTrademgener._trademgenService),
00085           _logOutputStream (iTrademgener._logOutputStream) {
00086     }
00087
00089     ~Trademgener() {
00090         _trademgenService = NULL;
00091         _logOutputStream = NULL;
00092     }
00093
00095     bool init (const std::string& iLogFilePath,
00096               const stdair::RandomSeed_T& iRandomSeed,
00097               const stdair::Filename_T& iDemandInputFilename,
00098               const std::string& iDBUser, const std::string& iDBPasswd,
00099               const std::string& iDBHost, const std::string& iDBPort,
00100               const std::string& iDBDBName) {
00101         bool isEverythingOK = true;
00102
00103         try {
00104
00105             // Check that the file path given as input corresponds to an actual
00106             file
00107             const bool isWriteable = (iLogFilePath.empty() == false);
00108             // stdair::BasFileMgr::isWriteable (iLogFilePath);
00109             if (isWriteable == false) {
00110                 isEverythingOK = false;
00111                 return isEverythingOK;
00112             }
00113
00114             // Set the log parameters
00115             _logOutputStream = new std::ofstream;

```

```

00115     assert (_logOutputStream != NULL);
00116
00117     // Open and clean the log outputfile
00118     _logOutputStream->open (iLogFilepath.c_str());
00119     _logOutputStream->clear();
00120
00121     // DEBUG
00122     *_logOutputStream << "Python wrapper initialisation" << std::endl;
00123     const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG,
00124                                           *_logOutputStream);
00125
00126     // Initialise the context
00127     stdair::BasDBParams lDBParams (iDBUser, iDBPasswd, iDBHost, iDBPort,
00128                                   iDBDBName);
00129     _trademgenService = new TRADEMGEN_Service (lLogParams, lDBParams,
00130                                               iRandomSeed);
00131     assert (_trademgenService != NULL);
00132
00133     // Create the DemandStream objects, and insert them within the BOM tree
00134     _trademgenService->parseAndLoad (iDemandInputFilename);
00135
00136     // DEBUG
00137     *_logOutputStream << "Python wrapper initialised" << std::endl;
00138
00139     } catch (const stdair::RootException& eTrademgenError) {
00140         *_logOutputStream << "Trademgen error: " << eTrademgenError.what()
00141             << std::endl;
00142     } catch (const std::exception& eStdError) {
00143         *_logOutputStream << "Error: " << eStdError.what() << std::endl;
00144     } catch (...) {
00145         *_logOutputStream << "Unknown error" << std::endl;
00146     }
00147     return isEverythingOK;
00148 }
00149
00150 private:
00151     TRADEMGEN_Service* _trademgenService;
00152     std::ofstream* _logOutputStream;
00153 };
00154
00155 ///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
00156 BOOST_PYTHON_MODULE(libpytrademgen) {
00157     boost::python::class_<TRADEMGEN::Trademgener> ("Trademgener")
00158         .def ("trademgen", &TRADEMGEN::Trademgener::trademgen)
00159         .def ("init", &TRADEMGEN::Trademgener::init);
00160 }

```

25.105 trademgen/service/TRADEMGEN_Service.cpp File Reference

```
#include <cassert>
```



```

#include <sstream>
#include <boost/make_shared.hpp>
#include <soci/soci.h>
#include <stdair/basic/BasChronometer.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <stdair/bom/BomRoot.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/bom/AirlineStruct.hpp>
#include <stdair/bom/EventStruct.hpp>
#include <stdair/bom/EventQueue.hpp>
#include <stdair/command/DBManagerForAirlines.hpp>
#include <stdair/service/Logger.hpp>
#include <stdair/service/DBSessionManager.hpp>
#include <stdair/STDAIR_Service.hpp>
#include <trademgen/basic/BasConst_TRADEMGEN_Service.hpp>
#include <trademgen/bom/BomDisplay.hpp>
#include <trademgen/bom/DemandStreamKey.hpp>
#include <trademgen/factory/FacTRADEMGENServiceContext.hpp>
#include <trademgen/command/DemandParser.hpp>
#include <trademgen/command/DemandManager.hpp>
#include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
#include <trademgen/TRADEMGEN_Service.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.106 TRADEMGEN_Service.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // Boost
00008 #include <boost/make_shared.hpp>
00009 #if defined(SOCI_HEADERS_BURIED)
00010 #include <soci/core/soci.h>
00011 #else // SOCI_HEADERS_BURIED
00012 #include <soci/soci.h>
00013 #endif // SOCI_HEADERS_BURIED
00014 // StdAir
00015 #include <stdair/basic/BasChronometer.hpp>
00016 #include <stdair/basic/BasConst_General.hpp>
00017 #include <stdair/bom/BomRoot.hpp>
00018 #include <stdair/bom/BookingRequestStruct.hpp>
00019 #include <stdair/bom/AirlineStruct.hpp>
00020 #include <stdair/bom/EventStruct.hpp>
00021 #include <stdair/bom/EventQueue.hpp>
00022 #include <stdair/command/DBManagerForAirlines.hpp>
00023 #include <stdair/service/Logger.hpp>
00024 #include <stdair/service/DBSessionManager.hpp>
00025 #include <stdair/STDAIR_Service.hpp>
00026 // TraDemGen
00027 #include <trademgen/basic/BasConst_TRADEMGEN_Service.hpp>
00028 #include <trademgen/bom/BomDisplay.hpp>
00029 #include <trademgen/bom/DemandStreamKey.hpp>
00030 #include <trademgen/factory/FacTRADEMGENServiceContext.hpp>
00031 #include <trademgen/command/DemandParser.hpp>
00032 #include <trademgen/command/DemandManager.hpp>
00033 #include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
00034 #include <trademgen/TRADEMGEN_Service.hpp>
00035
00036 namespace TRADEMGEN {
00037
00038 // //////////////////////////////////////
00039 TRADEMGEN_Service::TRADEMGEN_Service() : _trademgenServiceContext (NULL) {
00040     assert (false);

```

```

00041     }
00042
00043     // //////////////////////////////////////
00044     TRADEMGEN_Service::TRADEMGEN_Service (const TRADEMGEN_Service& iService)
00045     : _trademgenServiceContext (NULL) {
00046         assert (false);
00047     }
00048
00049     // //////////////////////////////////////
00050     TRADEMGEN_Service::TRADEMGEN_Service (const stdair::BasLogParams& iLogParams,
00051     const stdair::RandomSeed_T& iRandomSeed
00052 )
00053     : _trademgenServiceContext (NULL) {
00054
00055         // Initialise the STDAIR service handler
00056         stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00057             initStdAirService (iLogParams);
00058
00059         // Initialise the service context
00060         initServiceContext (iRandomSeed);
00061
00062         // Add the StdAir service context to the TRADEMGEN service context
00063         // \note TRADEMGEN owns the STDAIR service resources here.
00064         const bool ownStdairService = true;
00065         addStdAirService (lSTDAIR_Service_ptr, ownStdairService);
00066
00067         // Initialise the (remaining of the) context
00068         initTrademgenService();
00069     }
00070
00071     // //////////////////////////////////////
00072     TRADEMGEN_Service::TRADEMGEN_Service (const stdair::BasLogParams& iLogParams,
00073     const stdair::BasDBParams& iDBParams,
00074     const stdair::RandomSeed_T& iRandomSeed
00075 )
00076     : _trademgenServiceContext (NULL) {
00077
00078         // Initialise the STDAIR service handler
00079         stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00080             initStdAirService (iLogParams, iDBParams);
00081
00082         // Initialise the service context
00083         initServiceContext (iRandomSeed);
00084
00085         // Add the StdAir service context to the TRADEMGEN service context
00086         // \note TRADEMGEN owns the STDAIR service resources here.
00087         const bool ownStdairService = true;
00088         addStdAirService (lSTDAIR_Service_ptr, ownStdairService);
00089
00090         // Initialise the (remaining of the) context
00091         initTrademgenService();
00092     }
00093
00094     // //////////////////////////////////////
00095     TRADEMGEN_Service (stdair::STDAIR_ServicePtr_T ioSTDAIR_Service_ptr,
00096     const stdair::RandomSeed_T& iRandomSeed)
00097     : _trademgenServiceContext (NULL) {
00098
00099         // Initialise the service context
00100         initServiceContext (iRandomSeed);
00101
00102         // Add the StdAir service context to the TRADEMGEN service context
00103         // \note TraDemGen does not own the STDAIR service resources here.
00104         const bool doesNotOwnStdairService = false;
00105         addStdAirService (ioSTDAIR_Service_ptr, doesNotOwnStdairService);
00106
00107         // Initialise the context
00108         initTrademgenService();
00109     }
00110
00111     // //////////////////////////////////////
00112     TRADEMGEN_Service::~TRADEMGEN_Service () {
00113         // Delete/Clean all the objects from memory
00114         finalise();
00115     }
00116
00117     // //////////////////////////////////////
00118     void TRADEMGEN_Service::finalise() {
00119         assert (_trademgenServiceContext != NULL);
00120         // Reset the (Boost.)Smart pointer pointing on the STDAIR_Service object.
00121         _trademgenServiceContext->reset();
00122     }
00123
00124     // //////////////////////////////////////
00125     void TRADEMGEN_Service::
00126     initServiceContext (const stdair::RandomSeed_T& iRandomSeed) {

```

```

00126 // Initialise the service context
00127 TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00128     FacTRADEMGENServiceContext::instance().create (iRandomSeed);
00129 _trademgenServiceContext = &lTRADEMGEN_ServiceContext;
00130 }
00131
00132 // //////////////////////////////////////
00133 void TRADEMGEN_Service::
00134     addStdAirService (stdair::STDAIR_ServicePtr_T ioSTDAIR_Service_ptr,
00135                     const bool iOwnStdairService) {
00136     // Retrieve the TraDemGen service context
00137     assert (_trademgenServiceContext != NULL);
00138     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00139         *_trademgenServiceContext;
00140
00141     // Store the STDAIR service object within the (TRADEMGEN) service context
00142     lTRADEMGEN_ServiceContext.setSTDAIR_Service (ioSTDAIR_Service_ptr,
00143         iOwnStdairService);
00144 }
00145
00146 // //////////////////////////////////////
00147 stdair::STDAIR_ServicePtr_T TRADEMGEN_Service::
00148     initStdAirService (const stdair::BasLogParams& iLogParams,
00149                     const stdair::BasDBParams& iDBParams) {
00150
00151     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00152         boost::make_shared<stdair::STDAIR_Service> (iLogParams, iDBParams);
00153     assert (lSTDAIR_Service_ptr != NULL);
00154
00155     return lSTDAIR_Service_ptr;
00156 }
00157
00158 // //////////////////////////////////////
00159 stdair::STDAIR_ServicePtr_T TRADEMGEN_Service::
00160     initStdAirService (const stdair::BasLogParams& iLogParams) {
00161
00162     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00163         boost::make_shared<stdair::STDAIR_Service> (iLogParams);
00164     assert (lSTDAIR_Service_ptr != NULL);
00165
00166     return lSTDAIR_Service_ptr;
00167 }
00168
00169 // //////////////////////////////////////
00170 void TRADEMGEN_Service::initTrademgenService() {
00171     // Do nothing at this stage. A sample BOM tree may be built by
00172     // calling the buildSampleBom() method
00173 }
00174
00175 // //////////////////////////////////////
00176 void TRADEMGEN_Service::
00177     parseAndLoad (const stdair::Filename_T& iDemandInputFilename) {
00178
00179     // Retrieve the TraDemGen service context
00180     assert (_trademgenServiceContext != NULL);
00181     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00182         *_trademgenServiceContext;
00183
00184     // Retrieve the shared generator
00185     stdair::RandomGeneration& lSharedGenerator =
00186         lTRADEMGEN_ServiceContext.getUniformGenerator();
00187
00188     // Retrieve the default POS distribution
00189     const POSProbabilityMass_T& lDefaultPOSProbabilityMass =
00190         lTRADEMGEN_ServiceContext.getPOSProbabilityMass();
00191
00192     // Retrieve the StdAir service context
00193     stdair::STDAIR_Service& lSTDAIR_Service =
00194         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00195
00196     // Retrieve the event queue
00197     stdair::EventQueue& lEventQueue = lSTDAIR_Service.getEventQueue();
00198
00199     // Parse the input file and initialise the demand generators
00200     stdair::BasChronometer lDemandGeneration; lDemandGeneration.start();
00201     DemandParser::generateDemand (iDemandInputFilename, lEventQueue,
00202         lSharedGenerator, lDefaultPOSProbabilityMass)
00203 ;
00204
00205     const double lGenerationMeasure = lDemandGeneration.elapsed();
00206
00207     // DEBUG
00208     STDAIR_LOG_DEBUG ("Demand generation time: " << lGenerationMeasure);
00209 }
00210
00211 // //////////////////////////////////////
00212 void TRADEMGEN_Service::buildSampleBom() {
00213

```

```

00222 // Retrieve the TraDemGen service context
00223 if (_trademgenServiceContext == NULL) {
00224     throw stdair::NonInitialisedServiceException ("The TraDemGen service has
"
00225                                                     "not been initialised");
00226 }
00227 assert (_trademgenServiceContext != NULL);
00228
00229 // Retrieve the TraDemGen service context and whether it owns the Stdair
00230 // service
00231 TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00232     *_trademgenServiceContext;
00233 const bool doesOwnStdairService =
00234     lTRADEMGEN_ServiceContext.getOwnStdairServiceFlag();
00235
00236 // Retrieve the StdAIR service object from the (TraDemGen) service context
00237 stdair::STDAIR_Service& lSTDAIR_Service =
00238     lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00239
00240 if (doesOwnStdairService == true) {
00241     //
00242     lSTDAIR_Service.buildSampleBom();
00243 }
00244
00245 // Retrieve the shared generator
00246 stdair::RandomGeneration& lSharedGenerator =
00247     lTRADEMGEN_ServiceContext.getUniformGenerator();
00248
00249 // Retrieve the default POS distribution
00250 const POSProbabilityMass_T& lDefaultPOSProbabilityMass =
00251     lTRADEMGEN_ServiceContext.getPOSProbabilityMass();
00252
00253 // Retrieve the event queue
00254 stdair::EventQueue& lEventQueue = lSTDAIR_Service.getEventQueue();
00255
00256 // Delegate the BOM building to the dedicated service
00257 DemandManager::buildSampleBom (lEventQueue, lSharedGenerator,
00258     lDefaultPOSProbabilityMass);
00259 }
00260
00261 // //////////////////////////////////////
00262 stdair::BookingRequestStruct TRADEMGEN_Service::
00263 buildSampleBookingRequest (const bool isForCRS) {
00264
00265     // Retrieve the TraDemGen service context
00266     if (_trademgenServiceContext == NULL) {
00267         throw stdair::NonInitialisedServiceException ("The TraDemGen service has
"
00268                                                     "not been initialised");
00269     }
00270     assert (_trademgenServiceContext != NULL);
00271
00272     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00273         *_trademgenServiceContext;
00274
00275     // Retrieve the STDAIR service object from the (TraDemGen) service context
00276     stdair::STDAIR_Service& lSTDAIR_Service =
00277         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00278
00279     // Delegate the BOM building to the dedicated service
00280     return lSTDAIR_Service.buildSampleBookingRequest (isForCRS);
00281 }
00282
00283 // //////////////////////////////////////
00284 std::string TRADEMGEN_Service::csvDisplay() const {
00285
00286     // Retrieve the TraDemGen service context
00287     if (_trademgenServiceContext == NULL) {
00288         throw stdair::NonInitialisedServiceException ("The TraDemGen service has
"
00289                                                     "not been initialised");
00290     }
00291     assert (_trademgenServiceContext != NULL);
00292
00293     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00294         *_trademgenServiceContext;
00295
00296     // Retrieve the STDAIR service object from the (TraDemGen) service context
00297     stdair::STDAIR_Service& lSTDAIR_Service =
00298         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00299
00300     // Retrieve the event queue
00301     stdair::EventQueue& lEventQueue = lSTDAIR_Service.getEventQueue();
00302
00303     // Delegate the BOM building to the dedicated service
00304     return BomDisplay::csvDisplay (lEventQueue);
00305 }

```

```

00322
00323 ///////////////////////////////////////////////////////////////////
00324 void TRADEMGEN_Service::displayAirlineListFromDB() const {
00325
00326     // Retrieve the TraDemGen service context
00327     if (_trademgenServiceContext == NULL) {
00328         throw stdair::NonInitialisedServiceException ("The TraDemGen service has
"
00329                                                     "not been initialised");
00330     }
00331     assert (_trademgenServiceContext != NULL);
00332     // TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00333     // *_trademgenServiceContext;
00334
00335     // Get the date-time for the present time
00336     boost::posix_time::ptime lNowDateTime =
00337         boost::posix_time::second_clock::local_time();
00338     //boost::gregorian::date lNowDate = lNowDateTime.date();
00339
00340     // DEBUG
00341     STDAIR_LOG_DEBUG (std::endl
00342                     << "=====
00343                     << std::endl
00344                     << lNowDateTime);
00345
00346     // Delegate the query execution to the dedicated command
00347     stdair::BasChronometer lAirListChronometer;
00348     lAirListChronometer.start();
00349
00350     // Retrieve the database session handler
00351     stdair::DBSession_T& lDBSession =
00352         stdair::DBSessionManager::instance().getDBSession();
00353
00354     // Prepare and execute the select statement
00355     stdair::AirlineStruct lAirline;
00356     stdair::DBRequestStatement_T lSelectStatement (lDBSession);
00357     stdair::DBManagerForAirlines::prepareSelectStatement (lDBSession,
00358                                                         lSelectStatement,
00359                                                         lAirline);
00360
00361     // Prepare the SQL request corresponding to the select statement
00362     bool hasStillData = true;
00363     unsigned int idx = 0;
00364     while (hasStillData == true) {
00365         hasStillData =
00366             stdair::DBManagerForAirlines::iterateOnStatement (lSelectStatement,
00367                                                             lAirline);
00368
00369         // DEBUG
00370         STDAIR_LOG_DEBUG ("[" << idx << "]: " << lAirline);
00371
00372         // Iteration
00373         ++idx;
00374     }
00375
00376     const double lAirListMeasure = lAirListChronometer.elapsed();
00377
00378     // DEBUG
00379     STDAIR_LOG_DEBUG ("Sample service for airline list retrieval: "
00380                     << lAirListMeasure);
00381 }
00382
00383 ///////////////////////////////////////////////////////////////////
00384 const stdair::Count_T& TRADEMGEN_Service::
00385 getExpectedTotalNumberOfRequestsToBeGenerated() const {
00386
00387     // Retrieve the TraDemGen service context
00388     assert (_trademgenServiceContext != NULL);
00389     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00390         *_trademgenServiceContext;
00391
00392     // Retrieve the StdAir service context
00393     stdair::STDAIR_Service& lSTDAIR_Service =
00394         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00395
00396     // Retrieve the event queue object instance
00397     const stdair::EventQueue& lQueue = lSTDAIR_Service.getEventQueue();
00398
00399     // Delegate the call to the dedicated command
00400     const stdair::Count_T& oExpectedTotalNumberOfRequestsToBeGenerated =
00401         lQueue.getExpectedTotalNbOfEvents (stdair::EventType::BKG_REQ);
00402
00403     //
00404     return oExpectedTotalNumberOfRequestsToBeGenerated;
00405 }
00406
00407 ///////////////////////////////////////////////////////////////////

```

```

00408     const stdair::Count_T& TRADEMGEN_Service::
00409     getActualTotalNumberOfRequestsToBeGenerated() const {
00410
00411         // Retrieve the TraDemGen service context
00412         assert (_trademgenServiceContext != NULL);
00413         TRADEMGEN_ServiceContext& lTRADEMGGEN_ServiceContext =
00414             *_trademgenServiceContext;
00415
00416         // Retrieve the StdAir service context
00417         stdair::STDAIR_Service& lSTDAIR_Service =
00418             lTRADEMGGEN_ServiceContext.getSTDAIR_Service();
00419
00420         // Retrieve the event queue object instance
00421         const stdair::EventQueue& lQueue = lSTDAIR_Service.getEventQueue();
00422
00423         // Delegate the call to the dedicated command
00424         const stdair::Count_T& oActualTotalNumberOfRequestsToBeGenerated =
00425             lQueue.getActualTotalNbOfEvents (stdair::EventType::BKG_REQ);
00426
00427         //
00428         return oActualTotalNumberOfRequestsToBeGenerated;
00429     }
00430
00431     // //////////////////////////////////////
00432     const bool TRADEMGEN_Service::
00433     stillHavingRequestsToBeGenerated (const stdair::DemandStreamKeyStr_T& iKey,
00434                                     stdair::ProgressStatusSet& ioPSS,
00435                                     const stdair::DemandGenerationMethod&
00436                                     iDemandGenerationMethod) const {
00437
00438         // Retrieve the TraDemGen service context
00439         assert (_trademgenServiceContext != NULL);
00440         TRADEMGEN_ServiceContext& lTRADEMGGEN_ServiceContext =
00441             *_trademgenServiceContext;
00442
00443         // Retrieve the StdAir service context
00444         stdair::STDAIR_Service& lSTDAIR_Service =
00445             lTRADEMGGEN_ServiceContext.getSTDAIR_Service();
00446
00447         // Retrieve the event queue object instance
00448         const stdair::EventQueue& lQueue = lSTDAIR_Service.getEventQueue();
00449
00450         // Delegate the call to the dedicated command
00451         const bool oStillHavingRequestsToBeGenerated =
00452             DemandManager::stillHavingRequestsToBeGenerated (lQueue, iKey, ioPSS,
00453                                                             iDemandGenerationMethod)
00454 ;
00455
00456         //
00457         return oStillHavingRequestsToBeGenerated;
00458     }
00459
00460     // //////////////////////////////////////
00461     stdair::Count_T TRADEMGEN_Service::
00462     generateFirstRequests (const stdair::DemandGenerationMethod&
00463                           iDemandGenerationMethod) const {
00464
00465         // Retrieve the TraDemGen service context
00466         assert (_trademgenServiceContext != NULL);
00467         TRADEMGEN_ServiceContext& lTRADEMGGEN_ServiceContext =
00468             *_trademgenServiceContext;
00469
00470         // Retrieve the StdAir service context
00471         stdair::STDAIR_Service& lSTDAIR_Service =
00472             lTRADEMGGEN_ServiceContext.getSTDAIR_Service();
00473
00474         // Retrieve the event queue object instance
00475         stdair::EventQueue& lQueue = lSTDAIR_Service.getEventQueue();
00476
00477         // Retrieve the random generator
00478         stdair::RandomGeneration& lGenerator =
00479             lTRADEMGGEN_ServiceContext.getUniformGenerator();
00480
00481         // Delegate the call to the dedicated command
00482         const stdair::Count_T& oActualTotalNbOfEvents =
00483             DemandManager::generateFirstRequests (lQueue, lGenerator,
00484                                                  iDemandGenerationMethod);
00485
00486         //
00487         return oActualTotalNbOfEvents;
00488     }
00489
00490     // //////////////////////////////////////
00491     stdair::BookingRequestPtr_T TRADEMGEN_Service::
00492     generateNextRequest (const stdair::DemandStreamKeyStr_T& iKey,
00493                        const stdair::DemandGenerationMethod&
00494                        iDemandGenerationMethod) const {

```

```

00491
00492 // Retrieve the TraDemGen service context
00493 assert (_trademgenServiceContext != NULL);
00494 TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00495     *_trademgenServiceContext;
00496
00497 // Retrieve the StdAir service context
00498 stdair::STDAIR_Service& lSTDAIR_Service =
00499     lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00500
00501 // Retrieve the event queue object instance
00502 stdair::EventQueue& lQueue = lSTDAIR_Service.getEventQueue();
00503
00504 // Retrieve the random generator
00505 stdair::RandomGeneration& lGenerator =
00506     lTRADEMGEN_ServiceContext.getUniformGenerator();
00507
00508 // Delegate the call to the dedicated command
00509 return DemandManager::generateNextRequest (lQueue, lGenerator, iKey,
00510     iDemandGenerationMethod);
00511 }
00512
00513 // //////////////////////////////////////
00514 stdair::ProgressStatusSet TRADEMGEN_Service::
00515 popEvent (stdair::EventStruct& ioEventStruct) const {
00516
00517     // Retrieve the TraDemGen service context
00518     assert (_trademgenServiceContext != NULL);
00519     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00520         *_trademgenServiceContext;
00521
00522     // Retrieve the StdAir service context
00523     stdair::STDAIR_Service& lSTDAIR_Service =
00524         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00525
00526     // Retrieve the event queue object instance
00527     stdair::EventQueue& lQueue = lSTDAIR_Service.getEventQueue();
00528
00529     // Extract the next event from the queue
00530     return lQueue.popEvent (ioEventStruct);
00531 }
00532
00533 // //////////////////////////////////////
00534 bool TRADEMGEN_Service::isQueueDone() const {
00535
00536     // Retrieve the TraDemGen service context
00537     assert (_trademgenServiceContext != NULL);
00538     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00539         *_trademgenServiceContext;
00540
00541     // Retrieve the StdAir service context
00542     stdair::STDAIR_Service& lSTDAIR_Service =
00543         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00544
00545     // Retrieve the event queue object instance
00546     const stdair::EventQueue& lQueue = lSTDAIR_Service.getEventQueue();
00547
00548     // Calculates whether the event queue has been fully emptied
00549     const bool isQueueDone = lQueue.isQueueDone();
00550
00551     //
00552     return isQueueDone;
00553 }
00554
00555 // //////////////////////////////////////
00556 bool TRADEMGEN_Service::
00557 generateCancellation (const stdair::TravelSolutionStruct& iTravelSolution,
00558     const stdair::PartySize_T& iPartySize,
00559     const stdair::DateTime_T& iRequestTime,
00560     const stdair::Date_T& iDepartureDate) const {
00561
00562     // Retrieve the TraDemGen service context
00563     assert (_trademgenServiceContext != NULL);
00564     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00565         *_trademgenServiceContext;
00566
00567     // Retrieve the random generator
00568     stdair::RandomGeneration& lGenerator =
00569         lTRADEMGEN_ServiceContext.getUniformGenerator();
00570
00571     // Retrieve the StdAir service context
00572     stdair::STDAIR_Service& lSTDAIR_Service =
00573         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00574
00575     // Retrieve the event queue object instance
00576     stdair::EventQueue& lQueue = lSTDAIR_Service.getEventQueue();
00577

```

```

00578     return DemandManager::generateCancellation (lQueue, lGenerator,
00579                                                iTravelSolution, iPartySize,
00580                                                iRequestTime, iDepartureDate);
00581 }
00582
00583 // //////////////////////////////////////
00584 void TRADEMGEN_Service::reset() const {
00585
00586     // Retrieve the TraDemGen service context
00587     assert (_trademgenServiceContext != NULL);
00588     TRADEMGEN_ServiceContext& lTRADEMGEN_ServiceContext =
00589         *_trademgenServiceContext;
00590
00591     // Retrieve the StdAir service context
00592     stdair::STDAIR_Service& lSTDAIR_Service =
00593         lTRADEMGEN_ServiceContext.getSTDAIR_Service();
00594     // Retrieve the event queue object instance
00595     stdair::EventQueue& lQueue = lSTDAIR_Service.getEventQueue();
00596
00597     // Retrieve the shared generator
00598     stdair::RandomGeneration& lSharedGenerator =
00599         lTRADEMGEN_ServiceContext.getUniformGenerator();
00600
00601     // Delegate the call to the dedicated command
00602     DemandManager::reset (lQueue, lSharedGenerator.getBaseGenerator());
00603 }
00604 }

```

25.107 trademgen/service/TRADEMGEN_ServiceContext.cpp File Reference

```

#include <cassert>
#include <sstream>
#include <stdair/STDAIR_Service.hpp>
#include <stdair/basic/BasConst_General.hpp>
#include <trademgen/basic/BasConst_DemandGeneration.hpp>
#include <trademgen/service/TRADEMGEN_ServiceContext.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

25.108 TRADEMGEN_ServiceContext.cpp

```

00001 // //////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // StdAir
00008 #include <stdair/STDAIR_Service.hpp>
00009 #include <stdair/basic/BasConst_General.hpp>
00010 // TraDemGen
00011 #include <trademgen/basic/BasConst_DemandGeneration.hpp>
00012 #include <trademgen/service/TRADEMGEN_ServiceContext.hpp>
00013
00014 namespace TRADEMGEN {
00015
00016 // //////////////////////////////////////
00017 TRADEMGEN_ServiceContext::TRADEMGEN_ServiceContext ()
00018     : _ownStdairService (false), _uniformGenerator (stdair::DEFAULT_RANDOM_SEED
00019 ),
00020     _posProbabilityMass (DEFAULT_POS_PROBALILITY_MASS) {
00021
00022 // //////////////////////////////////////
00023 TRADEMGEN_ServiceContext::
00024 TRADEMGEN_ServiceContext (const TRADEMGEN_ServiceContext& iServiceContext)
00025     : _ownStdairService (false), _uniformGenerator (stdair::DEFAULT_RANDOM_SEED
00026 ),
00027     _posProbabilityMass (DEFAULT_POS_PROBALILITY_MASS) {
00028
00029 // //////////////////////////////////////

```



```

00030 TRADEMGEN_ServiceContext::
00031 TRADEMGEN_ServiceContext (const stdair::RandomSeed_T& iRandomSeed)
00032 : _ownStdairService (false), _uniformGenerator (iRandomSeed),
00033   _posProbabilityMass (DEFAULT_POS_PROBALILITY_MASS) {
00034 }
00035
00036 // //////////////////////////////////////
00037 TRADEMGEN_ServiceContext::~TRADEMGEN_ServiceContext() {
00038 }
00039
00040 // //////////////////////////////////////
00041 const std::string TRADEMGEN_ServiceContext::shortDisplay() const {
00042     std::ostringstream ostr;
00043     ostr << "TRADEMGEN_ServiceContext -- Owns StdAir service: "
00044           << _ownStdairService << " -- Generator: " << _uniformGenerator;
00045     return ostr.str();
00046 }
00047
00048 // //////////////////////////////////////
00049 const std::string TRADEMGEN_ServiceContext::display() const {
00050     std::ostringstream ostr;
00051     ostr << shortDisplay();
00052     return ostr.str();
00053 }
00054
00055 // //////////////////////////////////////
00056 const std::string TRADEMGEN_ServiceContext::describe() const {
00057     return shortDisplay();
00058 }
00059
00060 // //////////////////////////////////////
00061 void TRADEMGEN_ServiceContext::reset() {
00062     if (_ownStdairService == true) {
00063         _stdairService.reset();
00064     }
00065 }
00066
00067 }

```

25.109 trademgen/service/TRADEMGEN_ServiceContext.hpp File Reference

```

#include <string>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_service_types.hpp>
#include <stdair/basic/RandomGeneration.hpp>
#include <stdair/bom/BookingRequestTypes.hpp>
#include <stdair/service/ServiceAbstract.hpp>
#include <trademgen/TRADEMGEN_Types.hpp>
#include <trademgen/basic/DemandCharacteristicsTypes.hpp>

```

Classes

- class [TRADEMGEN::TRADEMGEN_ServiceContext](#)
Class holding the context of the Trademgen services.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)

25.110 TRADEMGEN_ServiceContext.hpp

```

00001 #ifndef __TRADEMGEN_SVC_TRADEMGENSEVICECONTEXT_HPP
00002 #define __TRADEMGEN_SVC_TRADEMGENSEVICECONTEXT_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section

```

```

00006 // //////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // StdAir
00010 #include <stdair/stdair_basic_types.hpp>
00011 #include <stdair/stdair_service_types.hpp>
00012 #include <stdair/basic/RandomGeneration.hpp>
00013 #include <stdair/bom/BookingRequestTypes.hpp>
00014 #include <stdair/service/ServiceAbstract.hpp>
00015 // TraDemGen
00016 #include <trademgen/TRADEMGEN_Types.hpp>
00017 #include <trademgen/basic/DemandCharacteristicsTypes.hpp>
00018
00019 // Forward declarations
00020 namespace stdair {
00021     struct DemandCharacteristics;
00022     struct DemandDistribution;
00023 }
00024
00025 namespace TRADEMGEN {
00026
00030     class TRADEMGEN_ServiceContext : public stdair::ServiceAbstract {
00036         friend class TRADEMGEN_Service;
00037         friend class FacTRADEMGENServiceContext;
00038
00039     private:
00040         // ////////// Getters //////////
00044         stdair::STDAIR_ServicePtr_T getSTDAIR_ServicePtr() const {
00045             return _stdairService;
00046         }
00047
00051         stdair::STDAIR_Service& getSTDAIR_Service() const {
00052             assert (_stdairService != NULL);
00053             return *_stdairService;
00054         }
00055
00059         const bool getOwnStdairServiceFlag() const {
00060             return _ownStdairService;
00061         }
00062
00066         stdair::RandomGeneration& getUniformGenerator() {
00067             return _uniformGenerator;
00068         }
00069
00073         const POSProbabilityMass_T& getPOSProbabilityMass() const {
00074             return _posProbabilityMass;
00075         }
00076
00077     private:
00078         // ////////// Setters //////////
00083         void setSTDAIR_Service (stdair::STDAIR_ServicePtr_T ioSTDAIR_ServicePtr,
00084                                 const bool iOwnStdairService) {
00085             _stdairService = ioSTDAIR_ServicePtr;
00086             _ownStdairService = iOwnStdairService;
00087         }
00088
00089     private:
00090         // ////////// Display Methods //////////
00091         const std::string shortDisplay() const;
00095
00096         const std::string display() const;
00100
00101         const std::string describe() const;
00106
00107     private:
00108
00110         TRADEMGEN_ServiceContext (const stdair::RandomSeed_T&);
00117         TRADEMGEN_ServiceContext ();
00121         TRADEMGEN_ServiceContext (const TRADEMGEN_ServiceContext&);
00122
00126         ~TRADEMGEN_ServiceContext ();
00127
00131         void reset();
00132
00133     private:
00134
00135         // ////////// Children //////////
00139         stdair::STDAIR_ServicePtr_T _stdairService;
00140
00144         bool _ownStdairService;
00145
00146     private:
00147         // ////////// Attributes //////////
00148

```

```

00155     stdair::RandomGeneration _uniformGenerator;
00156
00160     const POSProbabilityMass_T _posProbabilityMass;
00161 };
00162
00163 }
00164 #endif // __TRADEMGEN_SVC_TRADEMGENSEVICECONTEXT_HPP

```

25.111 trademgen/TRADEMGEN_Abstract.hpp File Reference

```

#include <istream>
#include <ostream>
#include <sstream>
#include <string>

```

Classes

- struct [TRADEMGEN::TRADEMGEN_Abstract](#)

Namespaces

- namespace [TRADEMGEN](#)

Functions

- `template<class charT, class traits > std::basic_ostream< charT, traits > &operator<< (std::basic_ostream< charT, traits > &ioOut, const TRADEMGEN::TRADEMGEN_Abstract &iStructure)`
- `template<class charT, class traits > std::basic_istream< charT, traits > &operator>> (std::basic_istream< charT, traits > &ioln, TRADEMGEN::TRADEMGEN_Abstract &iStructure)`

25.111.1 Function Documentation

25.111.1.1 `template<class charT, class traits > std::basic_ostream<charT, traits>& operator<< (std::basic_ostream< charT, traits > &ioOut, const TRADEMGEN::TRADEMGEN_Abstract &iStructure) [inline]`

Piece of code given by Nicolai M. Josuttis, Section 13.12.1 "Implementing Output Operators" (p653) of his book "The C++ Standard Library: A Tutorial and Reference", published by Addison-Wesley.

Definition at line 49 of file [TRADEMGEN_Abstract.hpp](#).

25.111.1.2 `template<class charT, class traits > std::basic_istream<charT, traits>& operator>> (std::basic_istream< charT, traits > &ioln, TRADEMGEN::TRADEMGEN_Abstract &iStructure) [inline]`

Piece of code given by Nicolai M. Josuttis, Section 13.12.1 "Implementing Output Operators" (pp655-657) of his book "The C++ Standard Library: A Tutorial and Reference", published by Addison-Wesley.

Definition at line 77 of file [TRADEMGEN_Abstract.hpp](#).

References [TRADEMGEN::TRADEMGEN_Abstract::fromStream\(\)](#).

25.112 TRADEMGEN_Abstract.hpp

```

00001 #ifndef __TRADEMGEN_TRADEMGEN_ABSTRACT_HPP

```

```

00002 #define __TRADEMGEN_TRADEMGEN_ABSTRACT_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <istream>
00009 #include <ostream>
00010 #include <sstream>
00011 #include <string>
00012
00013 namespace TRADEMGEN {
00014
00016     struct TRADEMGEN_Abstract {
00017     public:
00018         // ////////////////////////////////// Display support methods //////////////////////////////////
00021         virtual void toStream (std::ostream& ioOut) const = 0;
00022
00025         virtual void fromStream (std::istream& ioIn) = 0;
00026
00028         virtual std::string toString() const = 0;
00029
00030     protected:
00033         TRADEMGEN_Abstract () {}
00034         TRADEMGEN_Abstract (const TRADEMGEN_Abstract&) {}
00035
00037         virtual ~TRADEMGEN_Abstract() {}
00038     };
00039 }
00040
00046 template <class charT, class traits>
00047 inline
00048 std::basic_ostream<charT, traits>&
00049 operator<< (std::basic_ostream<charT, traits>& ioOut,
00050           const TRADEMGEN::TRADEMGEN_Abstract& iStructure) {
00056     std::basic_ostringstream<charT,traits> ostr;
00057     ostr.copyfmt (ioOut);
00058     ostr.width (0);
00059
00060     // Fill string stream
00061     iStructure.toStream (ostr);
00062
00063     // Print string stream
00064     ioOut << ostr.str();
00065
00066     return ioOut;
00067 }
00068
00074 template <class charT, class traits>
00075 inline
00076 std::basic_istream<charT, traits>&
00077 operator>> (std::basic_istream<charT, traits>& ioIn,
00078           TRADEMGEN::TRADEMGEN_Abstract& ioStructure) {
00079     // Fill Bom object with input stream
00080     ioStructure.fromStream (ioIn);
00081     return ioIn;
00082 }
00083
00084 #endif // __TRADEMGEN_TRADEMGEN_ABSTRACT_HPP

```

25.113 trademgen/TRADEMGEN_Exceptions.hpp File Reference

```

#include <exception>
#include <stdair/stdair_exceptions.hpp>

```

Classes

- class [TRADEMGEN::TrademgenGenerationException](#)
- class [TRADEMGEN::DemandInputFileNotFoundException](#)
- class [TRADEMGEN::IndexOutOfRangeException](#)

Namespaces

- namespace [TRADEMGEN](#)

25.114 TRADEMGEN_Exceptions.hpp

```

00001 #ifndef __TRADEMGEN_TRADEMGEN_EXCEPTIONS_HPP
00002 #define __TRADEMGEN_TRADEMGEN_EXCEPTIONS_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////
00007 // STL
00008 #include <exception>
00009 // StdAir
00010 #include <stdair/stdair_exceptions.hpp>
00011
00012 namespace TRADEMGEN {
00013
00014     // /////////////////////////////////// Exceptions ///////////////////////////////////
00018     class TrademgenGenerationException : public stdair::RootException {
00019     public:
00023         TrademgenGenerationException (const std::string& iWhat)
00024             : stdair::RootException (iWhat) {}
00025     };
00026
00030     class DemandInputFileNotFoundException
00031     : public stdair::FileNotFoundException {
00032     public:
00036         DemandInputFileNotFoundException (const std::string& iWhat)
00037             : stdair::FileNotFoundException (iWhat) {}
00038     };
00039
00043     class IndexOutOfRangeException : public TrademgenGenerationException {
00044     public:
00048         IndexOutOfRangeException (const std::string& iWhat)
00049             : TrademgenGenerationException (iWhat) {}
00050     };
00051
00052 }
00053 #endif // __TRADEMGEN_TRADEMGEN_EXCEPTIONS_HPP
00054

```

25.115 trademgen/TRADEMGEN.Service.hpp File Reference

```

#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_demand_types.hpp>
#include <stdair/stdair_maths_types.hpp>
#include <stdair/stdair_service_types.hpp>
#include <stdair/basic/DemandGenerationMethod.hpp>
#include <stdair/bom/BookingRequestTypes.hpp>
#include <stdair/bom/EventTypes.hpp>

```

Classes

- class [TRADEMGEN::TRADEMGEN_Service](#)
class holding the services related to Travel Demand Generation.

Namespaces

- namespace [stdair](#)
Forward declarations.
- namespace [TRADEMGEN](#)

25.116 TRADEMGEN_Service.hpp

```

00001 #ifndef __TRADEMGEN_TRADEMGEN_SERVICE_HPP
00002 #define __TRADEMGEN_TRADEMGEN_SERVICE_HPP
00003
00004 // //////////////////////////////////////
00005 // Import section

```

```

00006 // //////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/stdair_demand_types.hpp>
00010 #include <stdair/stdair_maths_types.hpp>
00011 #include <stdair/stdair_service_types.hpp>
00012 #include <stdair/basic/DemandGenerationMethod.hpp>
00013 #include <stdair/bom/BookingRequestTypes.hpp>
00014 #include <stdair/bom/EventTypes.hpp>
00015
00016 // Forward declarations
00017 namespace stdair {
00018     class EventQueue;
00019     struct ProgressStatusSet;
00020     struct BasLogParams;
00021     struct BasDBParams;
00022     struct BookingRequestStruct;
00023     struct DemandCharacteristics;
00024     struct DemandDistribution;
00025     struct EventStruct;
00026     struct TravelSolutionStruct;
00027 }
00028
00029 namespace TRADEMGEN {
00030
00031     class TRADEMGEN_ServiceContext;
00032     struct DemandStreamKey;
00033
00034     class TRADEMGEN_Service {
00035     public:
00036         // ////////////////////////////////// Constructors and Destructors //////////////////////////////////
00037         TRADEMGEN_Service (const stdair::BasLogParams&, const stdair::BasDBParams&,
00038             const stdair::RandomSeed_T&);
00039
00040         TRADEMGEN_Service (const stdair::BasLogParams&, const stdair::RandomSeed_T&
00041 );
00042
00043         TRADEMGEN_Service (stdair::STDAIR_ServicePtr_T, const stdair::RandomSeed_T&
00044 );
00045
00046         void parseAndLoad (const stdair::Filename_T& iDemandInputFilename);
00047
00048         ~TRADEMGEN_Service();
00049
00050     public:
00051         // ////////////////////////////////// Business support methods //////////////////////////////////
00052         void buildSampleBom();
00053
00054         stdair::BookingRequestStruct
00055         buildSampleBookingRequest (const bool isForCRS = false);
00056
00057         void displayAirlineListFromDB() const;
00058
00059         const stdair::Count_T& getExpectedTotalNumberOfRequestsToBeGenerated()
00060 const;
00061
00062         const stdair::Count_T& getActualTotalNumberOfRequestsToBeGenerated() const;
00063
00064         const bool
00065         stillHavingRequestsToBeGenerated (const stdair::DemandStreamKeyStr_T&,
00066             stdair::ProgressStatusSet&,
00067             const stdair::DemandGenerationMethod&)
00068 const;
00069
00070         stdair::Count_T
00071         generateFirstRequests (const stdair::DemandGenerationMethod&) const;
00072
00073         stdair::BookingRequestPtr_T
00074         generateNextRequest (const stdair::DemandStreamKeyStr_T&,
00075             const stdair::DemandGenerationMethod&) const;
00076
00077         stdair::ProgressStatusSet popEvent (stdair::EventStruct&) const;
00078
00079         bool isQueueDone() const;
00080
00081         bool generateCancellation (const stdair::TravelSolutionStruct&,
00082             const stdair::PartySize_T&,
00083             const stdair::DateTime_T&,
00084             const stdair::Date_T&) const;
00085
00086         void reset() const;
00087
00088     public:
00089         // ////////////////////////////////// Display support methods //////////////////////////////////
00090         std::string csvDisplay() const;

```

```

00362
00363
00364 private:
00365 // ////////////////////////////////// Constructors and Destructors //////////////////////////////////
00366 TRADEMGEN_Service();
00370
00374 TRADEMGEN_Service (const TRADEMGEN_Service&);
00375
00387 stdair::STDAIR_ServicePtr_T initStdAirService (const stdair::BasLogParams&,
00388                                               const stdair::BasDBParams&);
00389
00399 stdair::STDAIR_ServicePtr_T initStdAirService (const stdair::BasLogParams&)
;
00400
00409 void addStdAirService (stdair::STDAIR_ServicePtr_T ioSTDAIR_ServicePtr,
00410                       const bool iOwnStdairService);
00411
00418 void initServiceContext (const stdair::RandomSeed_T&);
00419
00426 void initTrademgenService();
00427
00431 void finalise();
00432
00433
00434 private:
00435 // ////////////////////////////////// Service Context //////////////////////////////////
00439 TRADEMGEN_ServiceContext* _trademgenServiceContext;
00440 };
00441
00442 }
00443 #endif // __TRADEMGEN_TRADEMGEN_SERVICE_HPP

```

25.117 trademgen/TRADEMGEN_Types.hpp File Reference

```

#include <boost/shared_ptr.hpp>
#include <trademgen/TRADEMGEN_Exceptions.hpp>

```

Namespaces

- namespace [TRADEMGEN](#)

Typedefs

- typedef boost::shared_ptr
< [TRADEMGEN_Service](#) > [TRADEMGEN::TRADEMGEN_ServicePtr_T](#)

25.118 TRADEMGEN_Types.hpp

```

00001 #ifndef __TRADEMGEN_TRADEMGEN_TYPES_HPP
00002 #define __TRADEMGEN_TRADEMGEN_TYPES_HPP
00003
00004 // //////////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////////
00007 // Boost
00008 #include <boost/shared_ptr.hpp>
00009 // TraDemGen
00010 #include <trademgen/TRADEMGEN_Exceptions.hpp>
00011
00012 namespace TRADEMGEN {
00013
00014 // Forward declarations
00015 class TRADEMGEN_Service;
00016
00017
00018 // ////////////////////////////////// Type definitions specific to DSim //////////////////////////////////
00022 typedef boost::shared_ptr<TRADEMGEN_Service> TRADEMGEN_ServicePtr_T;
00023
00024 }
00025 #endif // __TRADEMGEN_TRADEMGEN_TYPES_HPP
00026

```

25.119 trademgen/ui/qt/trademgen/main.cpp File Reference

```
#include <QtGui/QApplication>
#include "trademgen.h"
```

Functions

- int [main](#) (int argc, char **argv)

25.119.1 Function Documentation

25.119.1.1 int main (int argc, char ** argv)

Definition at line 5 of file [main.cpp](#).

25.120 main.cpp

```
00001 #include <QtGui/QApplication>
00002 #include "trademgen.h"
00003
00004
00005 int main(int argc, char** argv)
00006 {
00007     QApplication app(argc, argv);
00008     trademgen foo;
00009     foo.show();
00010     return app.exec();
00011 }
```


Index

- ~CategoricalAttribute
 - stdair::CategoricalAttribute, [66](#)
- ~CategoricalAttributeLite
 - TRADEMGEN::CategoricalAttributeLite, [67](#)
- ~ContinuousAttribute
 - TRADEMGEN::ContinuousAttribute, [70](#)
- ~ContinuousAttributeLite
 - TRADEMGEN::ContinuousAttributeLite, [71](#)
- ~DBParams
 - TRADEMGEN::DBParams, [74](#)
- ~DemandCharacteristics
 - TRADEMGEN::DemandCharacteristics, [84](#)
- ~DemandDistribution
 - TRADEMGEN::DemandDistribution, [87](#)
- ~DemandStream
 - TRADEMGEN::DemandStream, [96](#)
- ~DemandStreamKey
 - TRADEMGEN::DemandStreamKey, [105](#)
- ~DemandStruct
 - TRADEMGEN::DemandStruct, [108](#)
- ~FlagSaver
 - TRADEMGEN::FlagSaver, [117](#)
- ~RandomGenerationContext
 - TRADEMGEN::RandomGenerationContext, [121](#)
- ~Trademgener
 - TRADEMGEN::Trademgener, [168](#)
- _arrivalPattern
 - TRADEMGEN::DemandCharacteristics, [84](#)
- _channelProbDist
 - TRADEMGEN::DemandStruct, [109](#)
- _channelProbabilityMass
 - TRADEMGEN::DemandCharacteristics, [85](#)
- _dateRange
 - TRADEMGEN::DemandStruct, [109](#)
- _demand
 - TRADEMGEN::DemandParserHelper::Demand-Parser, [92](#)
 - TRADEMGEN::DemandParserHelper::doEnd-Demand, [114](#)
 - TRADEMGEN::DemandParserHelper::Parser-SemanticAction, [120](#)
 - TRADEMGEN::DemandParserHelper::store-ChannelCode, [124](#)
 - TRADEMGEN::DemandParserHelper::store-ChannelProbMass, [125](#)
 - TRADEMGEN::DemandParserHelper::store-DemandMean, [127](#)
 - TRADEMGEN::DemandParserHelper::store-DemandStdDev, [128](#)
 - TRADEMGEN::DemandParserHelper::store-Destination, [129](#)
 - TRADEMGEN::DemandParserHelper::storeDow, [131](#)
 - TRADEMGEN::DemandParserHelper::storeDTD, [132](#)
 - TRADEMGEN::DemandParserHelper::storeDTD-ProbMass, [134](#)
 - TRADEMGEN::DemandParserHelper::storeFF-Code, [135](#)
 - TRADEMGEN::DemandParserHelper::storeFF-ProbMass, [136](#)
 - TRADEMGEN::DemandParserHelper::storeOrigin, [138](#)
 - TRADEMGEN::DemandParserHelper::storePos-Code, [139](#)
 - TRADEMGEN::DemandParserHelper::storePos-ProbMass, [140](#)
 - TRADEMGEN::DemandParserHelper::storePref-Cabin, [142](#)
 - TRADEMGEN::DemandParserHelper::storePref-DepDateRangeEnd, [143](#)
 - TRADEMGEN::DemandParserHelper::storePref-DepDateRangeStart, [145](#)
 - TRADEMGEN::DemandParserHelper::storePref-DepTime, [146](#)
 - TRADEMGEN::DemandParserHelper::storePref-DepTimeProbMass, [147](#)
 - TRADEMGEN::DemandParserHelper::storeStay-Code, [149](#)
 - TRADEMGEN::DemandParserHelper::storeStay-ProbMass, [150](#)
 - TRADEMGEN::DemandParserHelper::storeTime-Value, [151](#)
 - TRADEMGEN::DemandParserHelper::storeTime-ValueProbMass, [153](#)
 - TRADEMGEN::DemandParserHelper::storeTrip-Code, [154](#)
 - TRADEMGEN::DemandParserHelper::storeTrip-ProbMass, [156](#)
 - TRADEMGEN::DemandParserHelper::storeWTP, [157](#)
- _demandCharacteristics
 - TRADEMGEN::DemandStream, [103](#)
- _demandCharacteristicsRandomGenerator
 - TRADEMGEN::DemandStream, [104](#)
- _demandDistribution
 - TRADEMGEN::DemandStream, [103](#)
- _demandMean
 - TRADEMGEN::DemandStruct, [109](#)
- _demandStdDev
 - TRADEMGEN::DemandStruct, [109](#)
- _describeKey
 - DemandGenerationTestSuite, [89](#)
- _destination
 - TRADEMGEN::DemandStruct, [109](#)
- _dow
 - TRADEMGEN::DemandStruct, [109](#)
- _dtdProbDist
 - TRADEMGEN::DemandStruct, [110](#)
- _eventQueue

- TRADEMGEN::DemandParserHelper::Demand-Parser, [92](#)
- TRADEMGEN::DemandParserHelper::doEnd-Demand, [114](#)
- _ffProbDist
 - TRADEMGEN::DemandStruct, [110](#)
- _frat5Pattern
 - TRADEMGEN::DemandCharacteristics, [85](#)
- _frequentFlyerProbabilityMass
 - TRADEMGEN::DemandCharacteristics, [85](#)
- _holderMap
 - TRADEMGEN::DemandStream, [103](#)
- _itChannelCode
 - TRADEMGEN::DemandStruct, [111](#)
- _itDTD
 - TRADEMGEN::DemandStruct, [112](#)
- _itDay
 - TRADEMGEN::DemandStruct, [111](#)
- _itFFCode
 - TRADEMGEN::DemandStruct, [112](#)
- _itHours
 - TRADEMGEN::DemandStruct, [111](#)
- _itMinutes
 - TRADEMGEN::DemandStruct, [111](#)
- _itMonth
 - TRADEMGEN::DemandStruct, [111](#)
- _itPosCode
 - TRADEMGEN::DemandStruct, [111](#)
- _itPrefDepTime
 - TRADEMGEN::DemandStruct, [112](#)
- _itSeconds
 - TRADEMGEN::DemandStruct, [111](#)
- _itStayDuration
 - TRADEMGEN::DemandStruct, [111](#)
- _itTimeValue
 - TRADEMGEN::DemandStruct, [112](#)
- _itTripCode
 - TRADEMGEN::DemandStruct, [111](#)
- _itYear
 - TRADEMGEN::DemandStruct, [110](#)
- _key
 - TRADEMGEN::DemandStream, [103](#)
- _meanNumberOfRequests
 - TRADEMGEN::DemandDistribution, [87](#)
- _minWTP
 - TRADEMGEN::DemandCharacteristics, [85](#)
 - TRADEMGEN::DemandStruct, [110](#)
- _origin
 - TRADEMGEN::DemandStruct, [109](#)
- _parent
 - TRADEMGEN::DemandStream, [103](#)
- _posProMass
 - TRADEMGEN::DemandStream, [104](#)
- _posProbDist
 - TRADEMGEN::DemandStruct, [109](#)
- _posProbabilityMass
 - TRADEMGEN::DemandCharacteristics, [84](#)
- TRADEMGEN::DemandParserHelper::Demand-Parser, [92](#)
- TRADEMGEN::DemandParserHelper::doEnd-Demand, [114](#)
- _prefCabin
 - TRADEMGEN::DemandStruct, [109](#)
- _prefDepDateEnd
 - TRADEMGEN::DemandStruct, [110](#)
- _prefDepDateStart
 - TRADEMGEN::DemandStruct, [110](#)
- _prefDepTimeProbDist
 - TRADEMGEN::DemandStruct, [110](#)
- _preferredDepartureTimeCumulativeDistribution
 - TRADEMGEN::DemandCharacteristics, [85](#)
- _randomGenerationContext
 - TRADEMGEN::DemandStream, [104](#)
- _requestDateTimeRandomGenerator
 - TRADEMGEN::DemandStream, [104](#)
- _stayDurationProbabilityMass
 - TRADEMGEN::DemandCharacteristics, [85](#)
- _stayProbDist
 - TRADEMGEN::DemandStruct, [110](#)
- _stdDevNumberOfRequests
 - TRADEMGEN::DemandDistribution, [87](#)
- _timeValueProbDist
 - TRADEMGEN::DemandStruct, [110](#)
- _totalNumberOfRequestsToBeGenerated
 - TRADEMGEN::DemandStream, [104](#)
- _tripProbDist
 - TRADEMGEN::DemandStruct, [110](#)
- _tripTypeProbabilityMass
 - TRADEMGEN::DemandCharacteristics, [85](#)
- _uniformGenerator
 - TRADEMGEN::DemandParserHelper::Demand-Parser, [92](#)
 - TRADEMGEN::DemandParserHelper::doEnd-Demand, [114](#)
- _valueOfTimeCumulativeDistribution
 - TRADEMGEN::DemandCharacteristics, [85](#)
- airline_code_p
 - TRADEMGEN::DemandParserHelper, [61](#)
- airport_p
 - TRADEMGEN::DemandParserHelper, [62](#)
- ArrivalPatternCumulativeDistribution_T
 - TRADEMGEN, [57](#)
- BINDIR
 - trademgen-paths.hpp, [267](#)
- BOOST_PYTHON_MODULE
 - pytrademgen.cpp, [272](#)
- batches/trademgen.cpp
 - generateDemand, [201](#)
 - main, [201](#)
 - NbOfRuns_T, [200](#)
 - operator<<, [201](#)
 - readConfiguration, [201](#)
 - stat_acc_type, [200](#)
 - stat_display, [200](#)

- BomAbstract, [63](#)
- bounded1_2_p_t
 - TRADEMGEN, [56](#)
- bounded1_3_p_t
 - TRADEMGEN, [56](#)
- bounded1_4_p_t
 - TRADEMGEN, [56](#)
- bounded2_p_t
 - TRADEMGEN, [56](#)
- bounded4_p_t
 - TRADEMGEN, [56](#)
- buildSampleBom
 - TRADEMGEN::TRADEMGEN_Service, [161](#)
- buildSampleBookingRequest
 - TRADEMGEN::TRADEMGEN_Service, [163](#)
- cabin_code_p
 - TRADEMGEN::DemandParserHelper, [62](#)
- CategoricalAttribute
 - stdair::CategoricalAttribute, [65](#)
- CategoricalAttributeLite
 - TRADEMGEN::CategoricalAttributeLite, [67](#)
- channel_code
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- channel_dist
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- channel_pair
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- channel_share
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- ChannelProbabilityMass_T
 - TRADEMGEN, [57](#)
- ChannelProbabilityMassFunction_T
 - TRADEMGEN, [57](#)
- char_t
 - TRADEMGEN, [55](#)
- check
 - TRADEMGEN::DBParams, [75](#)
- checkPOSValue
 - TRADEMGEN::DemandCharacteristics, [84](#)
- checkValue
 - TRADEMGEN::CategoricalAttributeLite, [68](#)
- chset_t
 - TRADEMGEN, [56](#)
- class_code_list_p
 - TRADEMGEN::DemandParserHelper, [62](#)
- CmdAbstract, [68](#)
- ContinuousAttribute
 - TRADEMGEN::ContinuousAttribute, [69](#)
- ContinuousAttributeLite
 - TRADEMGEN::ContinuousAttributeLite, [71](#)
- ContinuousDistribution_T
 - TRADEMGEN::ContinuousAttribute, [69](#)
 - TRADEMGEN::ContinuousAttributeLite, [71](#)
- ContinuousFloatDuration_T
 - TRADEMGEN, [57](#)
- ContinuousInverseDistribution_T
 - TRADEMGEN::ContinuousAttribute, [69](#)
- convertFloatIntoDuration
 - TRADEMGEN::DemandStream, [103](#)
- create
 - TRADEMGEN::FacTRADEMGENServiceContext, [116](#)
- createFRAT5Pattern
 - TRADEMGEN::DefaultMap, [76](#)
- createPOSProbMass
 - TRADEMGEN::DefaultMap, [76](#)
- createStringFromWordList
 - trademgen_with_db.cpp, [210](#)
- csvDisplay
 - TRADEMGEN::BomDisplay, [64](#)
 - TRADEMGEN::TRADEMGEN_Service, [166](#)
- CumulativeDistribution_T
 - TRADEMGEN, [58](#)
- DATADIR
 - trademgen-paths.hpp, [267](#)
- DATAROOTDIR
 - trademgen-paths.hpp, [267](#)
- DBParams
 - TRADEMGEN::DBParams, [74](#)
- DBParamsNameList_T
 - TRADEMGEN, [59](#)
- DOCDIR
 - trademgen-paths.hpp, [267](#)
- date
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [79](#)
- day_p
 - TRADEMGEN::DemandParserHelper, [61](#)
- definition
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [79](#)
- demand
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [79](#)
- demand_end
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [79](#)
- demand_list
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [79](#)
- demand_params
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- DemandCharacteristics
 - TRADEMGEN::DemandCharacteristics, [83](#), [84](#)
- DemandDistribution
 - TRADEMGEN::DemandDistribution, [86](#), [87](#)
- DemandFileParser
 - TRADEMGEN::DemandFileParser, [88](#)
- DemandGenerationTestSuite, [89](#)
 - _describeKey, [89](#)
 - DemandGenerationTestSuite, [89](#)

- DemandGenerationTestSuite, [89](#)
- simpleEventGeneration, [89](#)
- DemandInputFileNotFoundException
 - TRADEMGEN::DemandInputFileNotFoundException, [90](#)
- DemandParser
 - TRADEMGEN::DemandParserHelper::DemandParser, [92](#)
- DemandParserHelper::doEndDemand
 - TRADEMGEN::DemandManager, [91](#)
- DemandStream
 - TRADEMGEN::DemandStream, [96](#)
- DemandStreamKey
 - TRADEMGEN::DemandStreamKey, [105](#)
- DemandStreamList_T
 - TRADEMGEN, [58](#)
- DemandStreamMap_T
 - TRADEMGEN, [59](#)
- DemandStruct
 - TRADEMGEN::DemandStruct, [108](#)
- describe
 - TRADEMGEN::DemandCharacteristics, [84](#)
 - TRADEMGEN::DemandDistribution, [87](#)
 - TRADEMGEN::DemandStruct, [108](#)
 - TRADEMGEN::RandomGenerationContext, [122](#)
- describeKey
 - TRADEMGEN::DemandStream, [102](#)
- destination
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- determineInverseCumulativeDistributionFromCumulativeDistribution
 - TRADEMGEN::ContinuousAttribute, [70](#)
- determineInverseCumulativeDistributionFromProbabilityMassFunction
 - stdair::CategoricalAttribute, [66](#)
- DictionaryKey_T
 - TRADEMGEN, [58](#)
- display
 - TRADEMGEN::DemandDistribution, [87](#)
 - TRADEMGEN::DemandStream, [102](#)
- displayAirlineListFromDB
 - TRADEMGEN::TRADEMGEN_Service, [164](#)
- displayCumulativeDistribution
 - TRADEMGEN::ContinuousAttribute, [70](#)
 - TRADEMGEN::ContinuousAttributeLite, [72](#)
- displayInverseCumulativeDistribution
 - stdair::CategoricalAttribute, [66](#)
 - TRADEMGEN::ContinuousAttribute, [70](#)
- displayProbabilityMass
 - TRADEMGEN::CategoricalAttributeLite, [68](#)
- displayProbabilityMassFunction
 - stdair::CategoricalAttribute, [66](#)
- doEndDemand
 - TRADEMGEN::DemandParserHelper::doEndDemand, [114](#)
- doc/local/authors.doc, [169](#)
- doc/local/codingrules.doc, [169](#)
- doc/local/copyright.doc, [169](#)
- doc/local/documentation.doc, [169](#)
- doc/local/features.doc, [169](#)
- doc/local/help_wanted.doc, [169](#)
- doc/local/howto_release.doc, [169](#)
- doc/local/index.doc, [169](#)
- doc/local/installation.doc, [169](#)
- doc/local/linking.doc, [169](#)
- doc/local/test.doc, [169](#)
- doc/local/users_guide.doc, [169](#)
- doc/local/verification.doc, [169](#)
- doc/tutorial/tutorial.doc, [169](#)
- dow
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [79](#)
- dow_p
 - TRADEMGEN::DemandParserHelper, [61](#)
- dtd_dist
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [82](#)
- dtd_pair
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [82](#)
- dtd_share
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [82](#)
- EXEC_PREFIX
 - trademgen-paths.hpp, [267](#)
- FRAT5Pattern_T
 - TRADEMGEN, [58](#)
- FacServiceAbstract, [115](#)
- family_code_p
 - TRADEMGEN::DemandParserHelper, [63](#)
- ff_code
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [81](#)
- ff_dist
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [81](#)
- ff_pair
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [81](#)
- ff_share
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [81](#)
- ff_type_p
 - TRADEMGEN::DemandParserHelper, [62](#)
- FileNotFoundException, [117](#)
- FlagSaver
 - TRADEMGEN::FlagSaver, [117](#)
- flight_number_p
 - TRADEMGEN::DemandParserHelper, [61](#)
- FrequentFlyerProbabilityMass_T
 - TRADEMGEN, [58](#)
- FrequentFlyerProbabilityMassFunction_T
 - TRADEMGEN, [58](#)
- fromStream

- TRADEMGEN::DBParams, 75
- TRADEMGEN::DemandDistribution, 87
- TRADEMGEN::DemandStream, 102
- TRADEMGEN::DemandStreamKey, 106
- TRADEMGEN::TRADEMGEN_Abstract, 159
- generateCancellation
 - TRADEMGEN::TRADEMGEN_Service, 166
- generateChannel
 - TRADEMGEN::DemandStream, 100
- generateDemand
 - batches/trademgen.cpp, 201
 - TRADEMGEN::DemandFileParser, 88
 - TRADEMGEN::DemandParser, 93
- generateEvents.cpp
 - main, 174
- generateFirstRequests
 - TRADEMGEN::TRADEMGEN_Service, 165
- generateFrequentFlyer
 - TRADEMGEN::DemandStream, 101
- generateNextRequest
 - TRADEMGEN::DemandStream, 101
 - TRADEMGEN::TRADEMGEN_Service, 165
- generatePOS
 - TRADEMGEN::DemandStream, 100
- generatePreferredDepartureTime
 - TRADEMGEN::DemandStream, 101
- generateStayDuration
 - TRADEMGEN::DemandStream, 100
- generateTimeOfRequestPoissonProcess
 - TRADEMGEN::DemandStream, 99
- generateTimeOfRequestStatisticsOrder
 - TRADEMGEN::DemandStream, 100
- generateTripType
 - TRADEMGEN::DemandStream, 100
- generateValueOfTime
 - TRADEMGEN::DemandStream, 101
- generateWTP
 - TRADEMGEN::DemandStream, 101
- getActualTotalNumberOfRequestsToBeGenerated
 - TRADEMGEN::TRADEMGEN_Service, 164
- getCumulativeProbabilitySoFar
 - TRADEMGEN::RandomGenerationContext, 121
- getDBName
 - TRADEMGEN::DBParams, 75
- getDate
 - TRADEMGEN::DemandStruct, 108
- getDemandCharacteristics
 - TRADEMGEN::DemandStream, 97
- getDemandDistribution
 - TRADEMGEN::DemandStream, 97
- getDerivativeValue
 - TRADEMGEN::ContinuousAttributeLite, 72
- getDestination
 - TRADEMGEN::DemandStream, 96
 - TRADEMGEN::DemandStreamKey, 106
- getExpectedTotalNumberOfRequestsToBeGenerated
 - TRADEMGEN::TRADEMGEN_Service, 164
- getHolderMap
 - TRADEMGEN::DemandStream, 96
- getHost
 - TRADEMGEN::DBParams, 74
- getKey
 - TRADEMGEN::DemandStream, 96
- getMeanNumberOfRequests
 - TRADEMGEN::DemandStream, 97
- getNumberOfRequestsGeneratedSoFar
 - TRADEMGEN::DemandStream, 97
 - TRADEMGEN::RandomGenerationContext, 121
- getOrigin
 - TRADEMGEN::DemandStream, 96
 - TRADEMGEN::DemandStreamKey, 106
- getPOSProbabilityMass
 - TRADEMGEN::DemandStream, 97
- getPOSValue
 - TRADEMGEN::DemandCharacteristics, 84
- getParent
 - TRADEMGEN::DemandStream, 96
- getPassword
 - TRADEMGEN::DBParams, 74
- getPort
 - TRADEMGEN::DBParams, 75
- getPreferredCabin
 - TRADEMGEN::DemandStream, 96
 - TRADEMGEN::DemandStreamKey, 106
- getPreferredDepartureDate
 - TRADEMGEN::DemandStream, 96
 - TRADEMGEN::DemandStreamKey, 106
- getStdDevNumberOfRequests
 - TRADEMGEN::DemandStream, 97
- getTime
 - TRADEMGEN::DemandStruct, 108
- getTotalNumberOfRequestsToBeGenerated
 - TRADEMGEN::DemandStream, 97
- getUpperBound
 - TRADEMGEN::ContinuousAttributeLite, 72
- getUser
 - TRADEMGEN::DBParams, 74
- getValue
 - stdair::CategoricalAttribute, 66
 - TRADEMGEN::CategoricalAttributeLite, 68
 - TRADEMGEN::ContinuousAttribute, 70
 - TRADEMGEN::ContinuousAttributeLite, 72
- grammar, 118
- HTMLDIR
 - trademgen-paths.hpp, 267
- hours_p
 - TRADEMGEN::DemandParserHelper, 62
- INCLUDEDIR
 - trademgen-paths.hpp, 267
- INFODIR
 - trademgen-paths.hpp, 267
- incrementGeneratedRequestsCounter
 - TRADEMGEN::DemandStream, 99
 - TRADEMGEN::RandomGenerationContext, 122
- IndexOutOfRangeException

- TRADEMGEN::IndexOutOfRangeException, 118
- init
 - TRADEMGEN::Trademgener, 168
- instance
 - TRADEMGEN::FacTRADEMGENServiceContext, 116
- int1_p
 - TRADEMGEN::DemandParserHelper, 62
- int1_p_t
 - TRADEMGEN, 55
- InverseCumulativeDistribution_T
 - stdair::CategoricalAttribute, 65
- isQueueDone
 - TRADEMGEN::TRADEMGEN_Service, 166
- iterateOnStatement
 - TRADEMGEN::DBManager, 73
- iterator_t
 - TRADEMGEN, 55
- Key_T
 - TRADEMGEN::DemandStream, 95
- KeyAbstract, 119
- keyToValue
 - TRADEMGEN::DictionaryManager, 113
- LIBDIR
 - trademgen-paths.hpp, 267
- LIBEXECDIR
 - trademgen-paths.hpp, 267
- MANDIR
 - trademgen-paths.hpp, 267
- main
 - batches/trademgen.cpp, 201
 - generateEvents.cpp, 174
 - main.cpp, 290
 - trademgen_with_db.cpp, 210
- main.cpp
 - main, 290
- minutes_p
 - TRADEMGEN::DemandParserHelper, 62
- month_p
 - TRADEMGEN::DemandParserHelper, 61
- NbOfRuns_T
 - batches/trademgen.cpp, 200
- not_to_be_parsed
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, 79
- operator<<
 - batches/trademgen.cpp, 201
 - TRADEMGEN_Abstract.hpp, 285
 - trademgen_with_db.cpp, 210
- operator>>
 - TRADEMGEN_Abstract.hpp, 285
- operator()
 - TRADEMGEN::DemandParserHelper::doEndDemand, 114
- TRADEMGEN::DemandParserHelper::storeChannelCode, 124
- TRADEMGEN::DemandParserHelper::storeChannelProbMass, 125
- TRADEMGEN::DemandParserHelper::storeDemandMean, 126
- TRADEMGEN::DemandParserHelper::storeDemandStdDev, 128
- TRADEMGEN::DemandParserHelper::storeDestination, 129
- TRADEMGEN::DemandParserHelper::storeDow, 131
- TRADEMGEN::DemandParserHelper::storeDTD, 132
- TRADEMGEN::DemandParserHelper::storeDTDProbMass, 133
- TRADEMGEN::DemandParserHelper::storeFFCode, 135
- TRADEMGEN::DemandParserHelper::storeFFProbMass, 136
- TRADEMGEN::DemandParserHelper::storeOrigin, 138
- TRADEMGEN::DemandParserHelper::storePosCode, 139
- TRADEMGEN::DemandParserHelper::storePosProbMass, 140
- TRADEMGEN::DemandParserHelper::storePrefCabin, 142
- TRADEMGEN::DemandParserHelper::storePrefDepDateRangeEnd, 143
- TRADEMGEN::DemandParserHelper::storePrefDepDateRangeStart, 144
- TRADEMGEN::DemandParserHelper::storePrefDepTime, 146
- TRADEMGEN::DemandParserHelper::storePrefDepTimeProbMass, 147
- TRADEMGEN::DemandParserHelper::storeStayCode, 149
- TRADEMGEN::DemandParserHelper::storeStayProbMass, 150
- TRADEMGEN::DemandParserHelper::storeTimeValue, 151
- TRADEMGEN::DemandParserHelper::storeTimeValueProbMass, 153
- TRADEMGEN::DemandParserHelper::storeTripCode, 154
- TRADEMGEN::DemandParserHelper::storeTripProbMass, 155
- TRADEMGEN::DemandParserHelper::storeWTP, 157
- operator=
 - TRADEMGEN::CategoricalAttributeLite, 68
 - TRADEMGEN::ContinuousAttributeLite, 72
- origin
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, 79
- PACKAGE
 - trademgen-paths.hpp, 266

- PACKAGE_NAME
 - trademgen-paths.hpp, [266](#)
- PACKAGE_VERSION
 - trademgen-paths.hpp, [266](#)
- PDFDIR
 - trademgen-paths.hpp, [267](#)
- POSProbabilityMass_T
 - TRADEMGEN, [57](#)
- POSProbabilityMassFunction_T
 - TRADEMGEN, [57](#)
- PREFIXDIR
 - trademgen-paths.hpp, [267](#)
- parseAndLoad
 - TRADEMGEN::TRADEMGEN_Service, [161](#)
- ParserSemanticAction
 - TRADEMGEN::DemandParserHelper::ParserSemanticAction, [120](#)
- passenger_type_p
 - TRADEMGEN::DemandParserHelper, [62](#)
- popEvent
 - TRADEMGEN::TRADEMGEN_Service, [165](#)
- pos_code
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- pos_dist
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- pos_pair
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- pos_share
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- pref_cabin
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- pref_dep_date_range
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [79](#)
- pref_dep_time_dist
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [81](#)
- pref_dep_time_pair
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [81](#)
- pref_dep_time_share
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [82](#)
- PreferredDepartureTimeContinuousDistribution_T
 - TRADEMGEN, [58](#)
- PreferredDepartureTimeCumulativeDistribution_T
 - TRADEMGEN, [58](#)
- prepareSelectStatement
 - TRADEMGEN::DBManager, [73](#)
- ProbabilityMassFunction_T
 - stdair::CategoricalAttribute, [65](#)
 - TRADEMGEN::CategoricalAttributeLite, [67](#)
- pytrademgen.cpp
 - BOOST_PYTHON_MODULE, [272](#)
- RandomGenerationContext
 - TRADEMGEN::RandomGenerationContext, [121](#)
- readConfiguration
 - batches/trademgen.cpp, [201](#)
 - trademgen_with_db.cpp, [210](#)
- repeat_p_t
 - TRADEMGEN, [56](#)
- reset
 - TRADEMGEN::DemandStream, [102](#)
 - TRADEMGEN::RandomGenerationContext, [122](#)
 - TRADEMGEN::TRADEMGEN_Service, [166](#)
- retrieveAirline
 - TRADEMGEN::DBManager, [73](#)
- RootException, [122](#)
- rule_t
 - TRADEMGEN, [55](#)
- SBINDIR
 - trademgen-paths.hpp, [267](#)
- STDAIR_SAMPLE_DIR
 - trademgen-paths.hpp, [268](#)
- SYSCONFDIR
 - trademgen-paths.hpp, [267](#)
- scanner_t
 - TRADEMGEN, [55](#)
- seconds_p
 - TRADEMGEN::DemandParserHelper, [62](#)
- ServiceAbstract, [123](#)
- setAll
 - TRADEMGEN::DemandStream, [99](#)
- setBoolFirstDateTimeRequest
 - TRADEMGEN::DemandStream, [99](#)
- setCumulativeProbabilitySoFar
 - TRADEMGEN::RandomGenerationContext, [122](#)
- setDBName
 - TRADEMGEN::DBParams, [75](#)
- setDemandCharacteristics
 - TRADEMGEN::DemandStream, [98](#)
- setDemandCharacteristicsRandomGeneratorSeed
 - TRADEMGEN::DemandStream, [98](#)
- setDemandDistribution
 - TRADEMGEN::DemandStream, [98](#)
- setHost
 - TRADEMGEN::DBParams, [75](#)
- setNumberOfRequestsGeneratedSoFar
 - TRADEMGEN::DemandStream, [98](#)
 - TRADEMGEN::RandomGenerationContext, [122](#)
- setPOSProbabilityMass
 - TRADEMGEN::DemandStream, [99](#)
- setPassword
 - TRADEMGEN::DBParams, [75](#)
- setPort
 - TRADEMGEN::DBParams, [75](#)
- setRequestDateTimeRandomGeneratorSeed
 - TRADEMGEN::DemandStream, [98](#)
- setTotalNumberOfRequestsToBeGenerated
 - TRADEMGEN::DemandStream, [98](#)

setUser
 TRADEMGEN::DBParams, 75
 simpleEventGeneration
 DemandGenerationTestSuite, 89
 start
 TRADEMGEN::DemandParserHelper::Demand-
 Parser::definition, 79
 stat_acc_type
 batches/trademgen.cpp, 200
 stat_display
 batches/trademgen.cpp, 200
 stay_dist
 TRADEMGEN::DemandParserHelper::Demand-
 Parser::definition, 81
 stay_duration_p
 TRADEMGEN::DemandParserHelper, 62
 stay_pair
 TRADEMGEN::DemandParserHelper::Demand-
 Parser::definition, 81
 stay_share
 TRADEMGEN::DemandParserHelper::Demand-
 Parser::definition, 81
 StayDurationProbabilityMass_T
 TRADEMGEN, 57
 StayDurationProbabilityMassFunction_T
 TRADEMGEN, 57
 stdair, 52
 stdair::CategoricalAttribute
 ~CategoricalAttribute, 66
 CategoricalAttribute, 65
 determineInverseCumulativeDistributionFrom-
 ProbabilityMassFunction, 66
 displayInverseCumulativeDistribution, 66
 displayProbabilityMassFunction, 66
 getValue, 66
 InverseCumulativeDistribution_T, 65
 ProbabilityMassFunction_T, 65
 stdair::CategoricalAttribute< T >, 64
 stdair::FacBom
 TRADEMGEN::DemandStream, 103
 stdair::FacBomManager
 TRADEMGEN::DemandStream, 103
 stillHavingRequestsToBeGenerated
 TRADEMGEN::DemandStream, 99
 TRADEMGEN::TRADEMGEN_Service, 164
 storeChannelCode
 TRADEMGEN::DemandParserHelper::store-
 ChannelCode, 124
 storeChannelProbMass
 TRADEMGEN::DemandParserHelper::store-
 ChannelProbMass, 125
 storeDTD
 TRADEMGEN::DemandParserHelper::storeDTD,
 132
 storeDTDProbMass
 TRADEMGEN::DemandParserHelper::storeDTD-
 ProbMass, 133
 storeDemandMean
 TRADEMGEN::DemandParserHelper::store-
 DemandMean, 126
 storeDemandStdDev
 TRADEMGEN::DemandParserHelper::store-
 DemandStdDev, 128
 storeDestination
 TRADEMGEN::DemandParserHelper::store-
 Destination, 129
 storeDow
 TRADEMGEN::DemandParserHelper::storeDow,
 130
 storeFFCode
 TRADEMGEN::DemandParserHelper::storeFF-
 Code, 135
 storeFFProbMass
 TRADEMGEN::DemandParserHelper::storeFF-
 ProbMass, 136
 storeOrigin
 TRADEMGEN::DemandParserHelper::storeOrigin,
 137
 storePosCode
 TRADEMGEN::DemandParserHelper::storePos-
 Code, 139
 storePosProbMass
 TRADEMGEN::DemandParserHelper::storePos-
 ProbMass, 140
 storePrefCabin
 TRADEMGEN::DemandParserHelper::storePref-
 Cabin, 141
 storePrefDepDateRangeEnd
 TRADEMGEN::DemandParserHelper::storePref-
 DepDateRangeEnd, 143
 storePrefDepDateRangeStart
 TRADEMGEN::DemandParserHelper::storePref-
 DepDateRangeStart, 144
 storePrefDepTime
 TRADEMGEN::DemandParserHelper::storePref-
 DepTime, 146
 storePrefDepTimeProbMass
 TRADEMGEN::DemandParserHelper::storePref-
 DepTimeProbMass, 147
 storeStayCode
 TRADEMGEN::DemandParserHelper::storeStay-
 Code, 148
 storeStayProbMass
 TRADEMGEN::DemandParserHelper::storeStay-
 ProbMass, 150
 storeTimeValue
 TRADEMGEN::DemandParserHelper::storeTime-
 Value, 151
 storeTimeValueProbMass
 TRADEMGEN::DemandParserHelper::storeTime-
 ValueProbMass, 152
 storeTripCode
 TRADEMGEN::DemandParserHelper::storeTrip-
 Code, 154
 storeTripProbMass

- TRADEMGEN::DemandParserHelper::storeTrip-
ProbMass, 155
- storeWTP
 - TRADEMGEN::DemandParserHelper::storeWTP,
157
- StructAbstract, 158
- TRADEMGEN, 52
 - ArrivalPatternCumulativeDistribution_T, 57
 - bounded1_2_p_t, 56
 - bounded1_3_p_t, 56
 - bounded1_4_p_t, 56
 - bounded2_p_t, 56
 - bounded4_p_t, 56
 - ChannelProbabilityMass_T, 57
 - ChannelProbabilityMassFunction_T, 57
 - char_t, 55
 - chset_t, 56
 - ContinuousFloatDuration_T, 57
 - CumulativeDistribution_T, 58
 - DBParamsNameList_T, 59
 - DemandStreamList_T, 58
 - DemandStreamMap_T, 59
 - DictionaryKey_T, 58
 - FRAT5Pattern_T, 58
 - FrequentFlyerProbabilityMass_T, 58
 - FrequentFlyerProbabilityMassFunction_T, 58
 - int1_p_t, 55
 - iterator_t, 55
 - POSProbabilityMass_T, 57
 - POSProbabilityMassFunction_T, 57
 - PreferredDepartureTimeContinuousDistribution_T,
58
 - PreferredDepartureTimeCumulativeDistribution_T,
58
 - repeat_p_t, 56
 - rule_t, 55
 - scanner_t, 55
 - StayDurationProbabilityMass_T, 57
 - StayDurationProbabilityMassFunction_T, 57
 - TripTypeProbabilityMass_T, 57
 - TripTypeProbabilityMassFunction_T, 57
 - uint1_2_p_t, 56
 - uint1_3_p_t, 56
 - uint1_4_p_t, 56
 - uint2_p_t, 56
 - uint4_p_t, 56
 - ValueOfTimeContinuousDistribution_T, 58
 - ValueOfTimeCumulativeDistribution_T, 58
- TRADEMGEN::BomDisplay, 63
 - csvDisplay, 64
- TRADEMGEN::CategoricalAttributeLite
 - ~CategoricalAttributeLite, 67
 - CategoricalAttributeLite, 67
 - checkValue, 68
 - displayProbabilityMass, 68
 - getValue, 68
 - operator=, 68
 - ProbabilityMassFunction_T, 67
- TRADEMGEN::CategoricalAttributeLite< T >, 66
- TRADEMGEN::ContinuousAttribute
 - ~ContinuousAttribute, 70
 - ContinuousAttribute, 69
 - ContinuousDistribution_T, 69
 - ContinuousInverseDistribution_T, 69
 - determineInverseCumulativeDistributionFrom-
CumulativeDistribution, 70
 - displayCumulativeDistribution, 70
 - displayInverseCumulativeDistribution, 70
 - getValue, 70
- TRADEMGEN::ContinuousAttribute< T >, 68
- TRADEMGEN::ContinuousAttributeLite
 - ~ContinuousAttributeLite, 71
 - ContinuousAttributeLite, 71
 - ContinuousDistribution_T, 71
 - displayCumulativeDistribution, 72
 - getDerivativeValue, 72
 - getUpperBound, 72
 - getValue, 72
 - operator=, 72
- TRADEMGEN::ContinuousAttributeLite< T >, 70
- TRADEMGEN::DBManager, 72
 - iterateOnStatement, 73
 - prepareSelectStatement, 73
 - retrieveAirline, 73
 - updateAirlineInDB, 73
- TRADEMGEN::DBParams, 73
 - ~DBParams, 74
 - check, 75
 - DBParams, 74
 - fromStream, 75
 - getDBName, 75
 - getHost, 74
 - getPassword, 74
 - getPort, 75
 - getUser, 74
 - setDBName, 75
 - setHost, 75
 - setPassword, 75
 - setPort, 75
 - setUser, 75
 - toShortString, 76
 - toStream, 75
 - toString, 76
- TRADEMGEN::DefaultMap, 76
 - createFRAT5Pattern, 76
 - createPOSProbMass, 76
- TRADEMGEN::DemandCharacteristics, 82
 - ~DemandCharacteristics, 84
 - _arrivalPattern, 84
 - _channelProbabilityMass, 85
 - _frat5Pattern, 85
 - _frequentFlyerProbabilityMass, 85
 - _minWTP, 85
 - _posProbabilityMass, 84
 - _preferredDepartureTimeCumulativeDistribution,
85

- _stayDurationProbabilityMass, 85
 - _tripTypeProbabilityMass, 85
 - _valueOfTimeCumulativeDistribution, 85
 - checkPOSValue, 84
 - DemandCharacteristics, 83, 84
 - describe, 84
 - getPOSValue, 84
- TRADEMGEN::DemandDistribution, 86
 - ~DemandDistribution, 87
 - _meanNumberOfRequests, 87
 - _stdDevNumberOfRequests, 87
 - DemandDistribution, 86, 87
 - describe, 87
 - display, 87
 - fromStream, 87
- TRADEMGEN::DemandFileParser, 88
 - DemandFileParser, 88
 - generateDemand, 88
- TRADEMGEN::DemandInputFileNotFoundExceptio, 89
- TRADEMGEN::DemandManager, 90
 - DemandParserHelper::doEndDemand, 91
- TRADEMGEN::DemandParser, 92
 - generateDemand, 93
- TRADEMGEN::DemandParserHelper, 60
 - airline_code_p, 61
 - airport_p, 62
 - cabin_code_p, 62
 - class_code_list_p, 62
 - day_p, 61
 - dow_p, 61
 - family_code_p, 63
 - ff_type_p, 62
 - flight_number_p, 61
 - hours_p, 62
 - int1_p, 62
 - minutes_p, 62
 - month_p, 61
 - passenger_type_p, 62
 - seconds_p, 62
 - stay_duration_p, 62
 - uint1_2_p, 63
 - uint1_3_p, 63
 - uint1_4_p, 63
 - uint2_p, 62
 - uint4_p, 63
 - year_p, 61
- TRADEMGEN::DemandParserHelper::DemandParser, 91
 - _demand, 92
 - _eventQueue, 92
 - _posProbabilityMass, 92
 - _uniformGenerator, 92
 - DemandParser, 92
- TRADEMGEN::DemandParserHelper::DemandParser-
::definition
 - channel_code, 80
 - channel_dist, 80
 - channel_pair, 80
 - channel_share, 80
 - date, 79
 - definition, 79
 - demand, 79
 - demand_end, 79
 - demand_list, 79
 - demand_params, 80
 - destination, 80
 - dow, 79
 - dtd_dist, 82
 - dtd_pair, 82
 - dtd_share, 82
 - ff_code, 81
 - ff_dist, 81
 - ff_pair, 81
 - ff_share, 81
 - origin, 79
 - pos_code, 80
 - pos_dist, 80
 - pos_pair, 80
 - pos_share, 80
 - pref_cabin, 80
 - start, 79
 - stay_dist, 81
 - stay_pair, 81
 - stay_share, 81
 - time, 82
 - trip_code, 81
 - trip_dist, 80
 - trip_pair, 81
 - trip_share, 81
 - wtp, 82
- TRADEMGEN::DemandParserHelper::DemandParser-
::definition< ScannerT >, 77
- TRADEMGEN::DemandParserHelper::ParserSemantic-
Action, 119
 - _demand, 120
 - ParserSemanticAction, 120
- TRADEMGEN::DemandParserHelper::doEndDemand, 113
 - _demand, 114
 - _eventQueue, 114
 - _posProbabilityMass, 114
 - _uniformGenerator, 114
 - doEndDemand, 114
 - operator(), 114
- TRADEMGEN::DemandParserHelper::storeChannel-
Code, 123
 - _demand, 124
 - operator(), 124
 - storeChannelCode, 124
- TRADEMGEN::DemandParserHelper::storeChannel-
ProbMass, 124
 - _demand, 125
 - operator(), 125
 - storeChannelProbMass, 125
- TRADEMGEN::DemandParserHelper::storeDTD, 131

- [_demand](#), [132](#)
 - [operator\(\)](#), [132](#)
 - [storeDTD](#), [132](#)
- TRADEMGEN::DemandParserHelper::storeDTDProb-
Mass, [133](#)
 - [_demand](#), [134](#)
 - [operator\(\)](#), [133](#)
- TRADEMGEN::DemandParserHelper::storeDemand-
Mean, [126](#)
 - [_demand](#), [127](#)
 - [operator\(\)](#), [126](#)
 - [storeDemandMean](#), [126](#)
- TRADEMGEN::DemandParserHelper::storeDemand-
StdDev, [127](#)
 - [_demand](#), [128](#)
 - [operator\(\)](#), [128](#)
 - [storeDemandStdDev](#), [128](#)
- TRADEMGEN::DemandParserHelper::storeDestination, [129](#)
 - [_demand](#), [129](#)
 - [operator\(\)](#), [129](#)
 - [storeDestination](#), [129](#)
- TRADEMGEN::DemandParserHelper::storeDow, [130](#)
 - [_demand](#), [131](#)
 - [operator\(\)](#), [131](#)
 - [storeDow](#), [130](#)
- TRADEMGEN::DemandParserHelper::storeFFCode, [134](#)
 - [_demand](#), [135](#)
 - [operator\(\)](#), [135](#)
 - [storeFFCode](#), [135](#)
- TRADEMGEN::DemandParserHelper::storeFFProb-
Mass, [136](#)
 - [_demand](#), [136](#)
 - [operator\(\)](#), [136](#)
- TRADEMGEN::DemandParserHelper::storeOrigin, [137](#)
 - [_demand](#), [138](#)
 - [operator\(\)](#), [138](#)
 - [storeOrigin](#), [137](#)
- TRADEMGEN::DemandParserHelper::storePosCode, [138](#)
 - [_demand](#), [139](#)
 - [operator\(\)](#), [139](#)
 - [storePosCode](#), [139](#)
- TRADEMGEN::DemandParserHelper::storePosProb-
Mass, [140](#)
 - [_demand](#), [140](#)
 - [operator\(\)](#), [140](#)
 - [storePosProbMass](#), [140](#)
- TRADEMGEN::DemandParserHelper::storePrefCabin, [141](#)
 - [_demand](#), [142](#)
 - [operator\(\)](#), [142](#)
 - [storePrefCabin](#), [141](#)
- TRADEMGEN::DemandParserHelper::storePrefDep-
DateRangeEnd, [142](#)
 - [_demand](#), [143](#)
 - [operator\(\)](#), [143](#)
- TRADEMGEN::DemandParserHelper::storePrefDep-
DateRangeStart, [144](#)
 - [operator\(\)](#), [144](#)
- TRADEMGEN::DemandParserHelper::storePrefDep-
Time, [145](#)
 - [_demand](#), [146](#)
 - [operator\(\)](#), [146](#)
 - [storePrefDepTime](#), [146](#)
- TRADEMGEN::DemandParserHelper::storePrefDep-
TimeProbMass, [147](#)
 - [_demand](#), [147](#)
 - [operator\(\)](#), [147](#)
- TRADEMGEN::DemandParserHelper::storeStayCode, [148](#)
 - [_demand](#), [149](#)
 - [operator\(\)](#), [149](#)
 - [storeStayCode](#), [148](#)
- TRADEMGEN::DemandParserHelper::storeStayProb-
Mass, [149](#)
 - [_demand](#), [150](#)
 - [operator\(\)](#), [150](#)
 - [storeStayProbMass](#), [150](#)
- TRADEMGEN::DemandParserHelper::storeTimeValue, [151](#)
 - [_demand](#), [151](#)
 - [operator\(\)](#), [151](#)
 - [storeTimeValue](#), [151](#)
- TRADEMGEN::DemandParserHelper::storeTimeValue-
ProbMass, [152](#)
 - [_demand](#), [153](#)
 - [operator\(\)](#), [153](#)
- TRADEMGEN::DemandParserHelper::storeTripCode, [153](#)
 - [_demand](#), [154](#)
 - [operator\(\)](#), [154](#)
 - [storeTripCode](#), [154](#)
- TRADEMGEN::DemandParserHelper::storeTripProb-
Mass, [155](#)
 - [_demand](#), [156](#)
 - [operator\(\)](#), [155](#)
 - [storeTripProbMass](#), [155](#)
- TRADEMGEN::DemandParserHelper::storeWTP, [156](#)
 - [_demand](#), [157](#)
 - [operator\(\)](#), [157](#)
 - [storeWTP](#), [157](#)
- TRADEMGEN::DemandStream, [93](#)
 - [~DemandStream](#), [96](#)
 - [_demandCharacteristics](#), [103](#)
 - [_demandCharacteristicsRandomGenerator](#), [104](#)
 - [_demandDistribution](#), [103](#)
 - [_holderMap](#), [103](#)
 - [_key](#), [103](#)
 - [_parent](#), [103](#)
 - [_posProMass](#), [104](#)
 - [_randomGenerationContext](#), [104](#)
 - [_requestDateTimeRandomGenerator](#), [104](#)
 - [_totalNumberOfRequestsToBeGenerated](#), [104](#)
 - [convertFloatIntoDuration](#), [103](#)

- DemandStream, 96
- describeKey, 102
- display, 102
- fromStream, 102
- generateChannel, 100
- generateFrequentFlyer, 101
- generateNextRequest, 101
- generatePOS, 100
- generatePreferredDepartureTime, 101
- generateStayDuration, 100
- generateTimeOfRequestPoissonProcess, 99
- generateTimeOfRequestStatisticsOrder, 100
- generateTripType, 100
- generateValueOfTime, 101
- generateWTP, 101
- getDemandCharacteristics, 97
- getDemandDistribution, 97
- getDestination, 96
- getHolderMap, 96
- getKey, 96
- getMeanNumberOfRequests, 97
- getNumberOfRequestsGeneratedSoFar, 97
- getOrigin, 96
- getPOSProbabilityMass, 97
- getParent, 96
- getPreferredCabin, 96
- getPreferredDepartureDate, 96
- getStdDevNumberOfRequests, 97
- getTotalNumberOfRequestsToBeGenerated, 97
- incrementGeneratedRequestsCounter, 99
- Key_T, 95
- reset, 102
- setAll, 99
- setBoolFirstDateTimeRequest, 99
- setDemandCharacteristics, 98
- setDemandCharacteristicsRandomGeneratorSeed, 98
- setDemandDistribution, 98
- setNumberOfRequestsGeneratedSoFar, 98
- setPOSProbabilityMass, 99
- setRequestDateTimeRandomGeneratorSeed, 98
- setTotalNumberOfRequestsToBeGenerated, 98
- stdair::FacBom, 103
- stdair::FacBomManager, 103
- stillHavingRequestsToBeGenerated, 99
- toStream, 102
- toString, 102
- TRADEMGEN::DemandStreamKey, 105
 - ~DemandStreamKey, 105
 - DemandStreamKey, 105
 - fromStream, 106
 - getDestination, 106
 - getOrigin, 106
 - getPreferredCabin, 106
 - getPreferredDepartureDate, 106
 - toStream, 106
 - toString, 106
- TRADEMGEN::DemandStruct, 107
 - ~DemandStruct, 108
 - _channelProbDist, 109
 - _dateRange, 109
 - _demandMean, 109
 - _demandStdDev, 109
 - _destination, 109
 - _dow, 109
 - _dtdProbDist, 110
 - _ffProbDist, 110
 - _itChannelCode, 111
 - _itDTD, 112
 - _itDay, 111
 - _itFFCode, 112
 - _itHours, 111
 - _itMinutes, 111
 - _itMonth, 111
 - _itPosCode, 111
 - _itPrefDepTime, 112
 - _itSeconds, 111
 - _itStayDuration, 111
 - _itTimeValue, 112
 - _itTripCode, 111
 - _itYear, 110
 - _minWTP, 110
 - _origin, 109
 - _posProbDist, 109
 - _prefCabin, 109
 - _prefDepDateEnd, 110
 - _prefDepDateStart, 110
 - _prefDepTimeProbDist, 110
 - _stayProbDist, 110
 - _timeValueProbDist, 110
 - _tripProbDist, 110
 - DemandStruct, 108
 - describe, 108
 - getDate, 108
 - getTime, 108
- TRADEMGEN::DictionaryManager, 112
 - keyToValue, 113
 - valueToKey, 113
- TRADEMGEN::FacTRADEMGENServiceContext, 115
- TRADEMGEN::FlagSaver, 117
 - ~FlagSaver, 117
 - FlagSaver, 117
- TRADEMGEN::IndexOutOfRangeException, 118
 - IndexOutOfRangeException, 118
- TRADEMGEN::RandomGenerationContext, 120
 - ~RandomGenerationContext, 121
 - describe, 122
 - getCumulativeProbabilitySoFar, 121
 - getNumberOfRequestsGeneratedSoFar, 121
 - incrementGeneratedRequestsCounter, 122
 - RandomGenerationContext, 121
 - reset, 122
 - setCumulativeProbabilitySoFar, 122
 - setNumberOfRequestsGeneratedSoFar, 122
- TRADEMGEN::TRADEMGEN_Abstract, 158
- TRADEMGEN::TRADEMGEN_Service, 160

- reset, 166
- TRADEMGEN::TrademgenGenerationException, 168
 - TrademgenGenerationException, 169
- TRADEMGEN::Trademgener, 167
 - ~Trademgener, 168
 - init, 168
 - trademgen, 168
 - Trademgener, 168
- TRADEMGEN_Abstract.hpp
 - operator<<, 285
 - operator>>, 285
- TRADEMGEN_Service
 - TRADEMGEN::DemandManager, 91
- TRADEMGEN_ServicePtr_T
 - TRADEMGEN, 59
- test/ Directory Reference, 51
- test/trademgen/ Directory Reference, 52
- test/trademgen/DemandGenerationTestSuite.cpp, 170
- test/trademgen/DemandGenerationTestSuite.hpp, 173
- test/trademgen/generateEvents.cpp, 173, 174
- TestFixture, 158
- time
 - TRADEMGEN::DemandParserHelper::Demand-
Parser::definition, 82
- time_value_dist
 - TRADEMGEN::DemandParserHelper::Demand-
Parser::definition, 82
- time_value_pair
 - TRADEMGEN::DemandParserHelper::Demand-
Parser::definition, 82
- time_value_share
 - TRADEMGEN::DemandParserHelper::Demand-
Parser::definition, 82
- toShortString
 - TRADEMGEN::DBParams, 76
- toStream
 - TRADEMGEN::DBParams, 75
 - TRADEMGEN::DemandStream, 102
 - TRADEMGEN::DemandStreamKey, 106
 - TRADEMGEN::TRADEMGEN_Abstract, 159
- toString
 - TRADEMGEN::DBParams, 76
 - TRADEMGEN::DemandStream, 102
 - TRADEMGEN::DemandStreamKey, 106
 - TRADEMGEN::TRADEMGEN_Abstract, 159
- tokeniseStringIntoWordList
 - trademgen_with_db.cpp, 210
- trademgen
 - TRADEMGEN::Trademgener, 168
- trademgen-paths.hpp
 - BINDIR, 267
 - DATADIR, 267
 - DATAROOTDIR, 267
 - DOCDIR, 267
 - EXEC_PREFIX, 267
 - HTMLDIR, 267
 - INCLUDEDIR, 267
 - INFODIR, 267
 - LIBDIR, 267
 - LIBEXECDIR, 267
 - MANDIR, 267
 - PACKAGE, 266
 - PACKAGE_NAME, 266
 - PACKAGE_VERSION, 266
 - PDFDIR, 267
 - PREFIXDIR, 267
 - SBINDIR, 267
 - STDAIR_SAMPLE_DIR, 268
 - SYSCONFDIR, 267
- trademgen/ Directory Reference, 51
- trademgen/DBParams.hpp, 268
- trademgen/TRADEMGEN_Abstract.hpp, 285
- trademgen/TRADEMGEN_Exceptions.hpp, 286, 287
- trademgen/TRADEMGEN_Service.hpp, 287
- trademgen/TRADEMGEN_Types.hpp, 289
- trademgen/basic/ Directory Reference, 49
- trademgen/basic/BasConst.cpp, 175
- trademgen/basic/BasConst_DemandGeneration.hpp, 176
- trademgen/basic/BasConst_TRADEMGEN_Service.-
hpp, 177
- trademgen/basic/BasParserTypes.hpp, 177, 179
- trademgen/basic/CategoricalAttribute.hpp, 180
- trademgen/basic/CategoricalAttributeLite.hpp, 182
- trademgen/basic/ContinuousAttribute.hpp, 184
- trademgen/basic/ContinuousAttributeLite.hpp, 186
- trademgen/basic/DemandCharacteristics.cpp, 189
- trademgen/basic/DemandCharacteristics.hpp, 191
- trademgen/basic/DemandCharacteristicsTypes.hpp, 192, 193
- trademgen/basic/DemandDistribution.cpp, 194
- trademgen/basic/DemandDistribution.hpp, 195
- trademgen/basic/DictionaryManager.cpp, 196
- trademgen/basic/DictionaryManager.hpp, 196, 197
- trademgen/basic/RandomGenerationContext.cpp, 197
- trademgen/basic/RandomGenerationContext.hpp, 198
- trademgen/batches/ Directory Reference, 50
- trademgen/batches/trademgen.cpp, 199, 202
- trademgen/batches/trademgen_with_db.cpp, 208, 211
- trademgen/bom/ Directory Reference, 50
- trademgen/bom/BomDisplay.cpp, 215
- trademgen/bom/BomDisplay.hpp, 216, 217
- trademgen/bom/DemandStream.cpp, 217
- trademgen/bom/DemandStream.hpp, 224, 225
- trademgen/bom/DemandStreamKey.cpp, 228
- trademgen/bom/DemandStreamKey.hpp, 229
- trademgen/bom/DemandStreamTypes.hpp, 230
- trademgen/bom/DemandStruct.cpp, 231
- trademgen/bom/DemandStruct.hpp, 233
- trademgen/command/ Directory Reference, 50
- trademgen/command/DBManager.cpp, 234, 235
- trademgen/command/DBManager.hpp, 237
- trademgen/command/DemandManager.cpp, 238
- trademgen/command/DemandManager.hpp, 248
- trademgen/command/DemandParser.cpp, 250
- trademgen/command/DemandParser.hpp, 251

- trademgen/command/DemandParserHelper.cpp, [251](#), [252](#)
- trademgen/command/DemandParserHelper.hpp, [262](#), [263](#)
- trademgen/config/ Directory Reference, [50](#)
- trademgen/config/trademgen-paths.hpp, [266](#), [268](#)
- trademgen/factory/ Directory Reference, [50](#)
- trademgen/factory/FacTRADEMGENSEerviceContext.cpp, [270](#)
- trademgen/factory/FacTRADEMGENSEerviceContext.hpp, [271](#)
- trademgen/python/ Directory Reference, [50](#)
- trademgen/python/pytrademgen.cpp, [272](#)
- trademgen/service/ Directory Reference, [51](#)
- trademgen/service/TRADEMGEN_Service.cpp, [274](#), [275](#)
- trademgen/service/TRADEMGEN_ServiceContext.cpp, [282](#)
- trademgen/service/TRADEMGEN_ServiceContext.hpp, [283](#)
- trademgen/ui/ Directory Reference, [52](#)
- trademgen/ui/qt/ Directory Reference, [51](#)
- trademgen/ui/qt/trademgen/ Directory Reference, [51](#)
- trademgen/ui/qt/trademgen/main.cpp, [290](#)
- trademgen/ui/qt/trademgen/trademgen.cpp, [208](#)
- trademgen_with_db.cpp
 - createStringFromWordList, [210](#)
 - main, [210](#)
 - operator<<, [210](#)
 - readConfiguration, [210](#)
 - tokeniseStringIntoWordList, [210](#)
 - WordList_T, [209](#)
- TrademgenGenerationException
 - TRADEMGEN::TrademgenGenerationException, [169](#)
- Trademgener
 - TRADEMGEN::Trademgener, [168](#)
- trip_code
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [81](#)
- trip_dist
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [80](#)
- trip_pair
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [81](#)
- trip_share
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [81](#)
- TripTypeProbabilityMass_T
 - TRADEMGEN, [57](#)
- TripTypeProbabilityMassFunction_T
 - TRADEMGEN, [57](#)
- uint1_2_p
 - TRADEMGEN::DemandParserHelper, [63](#)
- uint1_2_p_t
 - TRADEMGEN, [56](#)
- uint1_3_p
 - TRADEMGEN::DemandParserHelper, [63](#)
- uint1_3_p_t
 - TRADEMGEN, [56](#)
- uint1_4_p
 - TRADEMGEN::DemandParserHelper, [63](#)
- uint1_4_p_t
 - TRADEMGEN, [56](#)
- uint2_p
 - TRADEMGEN::DemandParserHelper, [62](#)
- uint2_p_t
 - TRADEMGEN, [56](#)
- uint4_p
 - TRADEMGEN::DemandParserHelper, [63](#)
- uint4_p_t
 - TRADEMGEN, [56](#)
- updateAirlineInDB
 - TRADEMGEN::DBManager, [73](#)
- ValueOfTimeContinuousDistribution_T
 - TRADEMGEN, [58](#)
- ValueOfTimeCumulativeDistribution_T
 - TRADEMGEN, [58](#)
- valueToKey
 - TRADEMGEN::DictionaryManager, [113](#)
- WordList_T
 - trademgen_with_db.cpp, [209](#)
- wtp
 - TRADEMGEN::DemandParserHelper::DemandParser::definition, [82](#)
- year_p
 - TRADEMGEN::DemandParserHelper, [61](#)