

BRAT

3.3.0

Generated by Doxygen 1.8.1.2

Tue Aug 4 2015 02:34:36

Contents

1	Module Index	1
1.1	Modules	1
2	Class Index	1
2.1	Class Hierarchy	1
3	Class Index	5
3.1	Class List	6
4	File Index	10
4.1	File List	10
5	Module Documentation	16
5.1	Error codes	16
5.1.1	Detailed Description	17
5.1.2	Macro Definition Documentation	17
5.2	Date error codes	18
5.2.1	Detailed Description	18
5.3	Cycle/date conversion error codes	19
5.3.1	Detailed Description	19
5.4	Algorithms classes	20
5.4.1	Detailed Description	22
5.4.2	Function Documentation	22
5.5	Tools	26
5.5.1	Detailed Description	42
5.5.2	Macro Definition Documentation	42
5.5.3	Typedef Documentation	42
5.5.4	Function Documentation	42
5.5.5	Variable Documentation	65
5.6	Criteria	66
5.6.1	Detailed Description	76
5.6.2	Function Documentation	76
5.6.3	Variable Documentation	82
5.7	Date conversion classes	83
5.7.1	Detailed Description	83
5.8	Errors management	84
5.8.1	Detailed Description	84
5.9	File services	85
5.9.1	Detailed Description	85
5.10	Parameters	86

5.10.1	Detailed Description	86
5.10.2	Function Documentation	86
5.11	Date conversion C APIs	88
5.11.1	Detailed Description	88
5.11.2	Function Documentation	88
5.12	C API for reading data	95
5.12.1	Detailed Description	95
5.12.2	Function Documentation	95
6	Class Documentation	97
6.1	_structDateDSM Struct Reference	97
6.1.1	Detailed Description	97
6.1.2	Member Data Documentation	97
6.2	_structDateJulian Struct Reference	97
6.2.1	Detailed Description	98
6.2.2	Member Data Documentation	98
6.3	_structDateSecond Struct Reference	98
6.3.1	Detailed Description	98
6.3.2	Member Data Documentation	98
6.4	_structDateYMDHMSM Struct Reference	99
6.4.1	Detailed Description	99
6.5	bratl::CAlgorithmException Class Reference	99
6.5.1	Detailed Description	100
6.5.2	Constructor & Destructor Documentation	100
6.6	bratl::CBratAlgoFilterGaussian1D Class Reference	101
6.6.1	Detailed Description	101
6.6.2	Constructor & Destructor Documentation	101
6.6.3	Member Function Documentation	102
6.7	bratl::CBratAlgoFilterGaussian2D Class Reference	102
6.7.1	Detailed Description	103
6.7.2	Constructor & Destructor Documentation	103
6.7.3	Member Function Documentation	103
6.8	bratl::CBratAlgoFilterLanczos1D Class Reference	104
6.8.1	Detailed Description	105
6.8.2	Constructor & Destructor Documentation	105
6.8.3	Member Function Documentation	105
6.9	bratl::CBratAlgoFilterLanczos2D Class Reference	106
6.9.1	Detailed Description	106
6.9.2	Constructor & Destructor Documentation	106
6.9.3	Member Function Documentation	107

6.10	bratl::CBratAlgoFilterLoess1D Class Reference	107
6.10.1	Detailed Description	109
6.10.2	Constructor & Destructor Documentation	109
6.10.3	Member Function Documentation	109
6.11	bratl::CBratAlgoFilterLoess2D Class Reference	111
6.11.1	Detailed Description	112
6.11.2	Constructor & Destructor Documentation	112
6.11.3	Member Function Documentation	112
6.12	bratl::CBratAlgoFilterMedian1D Class Reference	114
6.12.1	Detailed Description	115
6.12.2	Constructor & Destructor Documentation	115
6.12.3	Member Function Documentation	115
6.13	bratl::CBratAlgoFilterMedian2D Class Reference	117
6.13.1	Detailed Description	118
6.13.2	Constructor & Destructor Documentation	118
6.13.3	Member Function Documentation	118
6.14	bratl::CBratAlgorithmBase Class Reference	120
6.14.1	Detailed Description	123
6.14.2	Constructor & Destructor Documentation	123
6.14.3	Member Function Documentation	123
6.15	bratl::CBratAlgorithmGeosVel Class Reference	125
6.15.1	Detailed Description	127
6.15.2	Constructor & Destructor Documentation	127
6.15.3	Member Function Documentation	128
6.16	bratl::CBratAlgorithmGeosVelAtp Class Reference	128
6.16.1	Detailed Description	130
6.16.2	Constructor & Destructor Documentation	130
6.16.3	Member Function Documentation	130
6.17	bratl::CBratAlgorithmGeosVelGrid Class Reference	132
6.17.1	Detailed Description	134
6.18	bratl::CBratAlgorithmGeosVelGridU Class Reference	134
6.18.1	Detailed Description	136
6.19	bratl::CBratAlgorithmGeosVelGridV Class Reference	136
6.19.1	Detailed Description	137
6.20	bratl::CCriteria Class Reference	137
6.20.1	Detailed Description	138
6.20.2	Member Enumeration Documentation	138
6.20.3	Member Function Documentation	138
6.21	bratl::CCriteriaCycle Class Reference	139
6.21.1	Detailed Description	141

6.21.2	Constructor & Destructor Documentation	141
6.21.3	Member Function Documentation	141
6.21.4	Member Data Documentation	143
6.22	bratl::CCriteriaCycleInfo Class Reference	144
6.22.1	Detailed Description	145
6.23	bratl::CCriteriaDatetime Class Reference	145
6.23.1	Detailed Description	147
6.23.2	Constructor & Destructor Documentation	147
6.23.3	Member Function Documentation	148
6.23.4	Member Data Documentation	150
6.24	bratl::CCriteriaDatetimeInfo Class Reference	150
6.24.1	Detailed Description	151
6.25	bratl::CCriterialInfo Class Reference	152
6.25.1	Detailed Description	153
6.26	bratl::CCriteriaLatLon Class Reference	153
6.26.1	Detailed Description	154
6.26.2	Constructor & Destructor Documentation	155
6.26.3	Member Function Documentation	156
6.26.4	Member Data Documentation	158
6.27	bratl::CCriteriaLatLonInfo Class Reference	159
6.27.1	Detailed Description	160
6.28	bratl::CCriteriaPass Class Reference	160
6.28.1	Detailed Description	162
6.29	bratl::CCriteriaPassInfo Class Reference	162
6.29.1	Detailed Description	163
6.30	bratl::CCriteriaPassInt Class Reference	164
6.30.1	Detailed Description	166
6.31	bratl::CCriteriaPassIntInfo Class Reference	166
6.31.1	Detailed Description	167
6.32	bratl::CCriteriaPassString Class Reference	167
6.32.1	Detailed Description	169
6.33	bratl::CCriteriaPassStringInfo Class Reference	170
6.33.1	Detailed Description	171
6.34	bratl::CDataSet Class Reference	171
6.34.1	Detailed Description	172
6.34.2	Member Function Documentation	173
6.35	bratl::CDate Class Reference	173
6.35.1	Detailed Description	176
6.35.2	Constructor & Destructor Documentation	176
6.35.3	Member Function Documentation	177

6.35.4	Member Data Documentation	189
6.36	brathl::CDatePeriod Class Reference	190
6.36.1	Detailed Description	191
6.36.2	Constructor & Destructor Documentation	191
6.36.3	Member Function Documentation	192
6.36.4	Member Data Documentation	195
6.37	brathl::CDoubleArray Class Reference	195
6.37.1	Detailed Description	196
6.38	brathl::CDoubleMap Class Reference	196
6.38.1	Detailed Description	196
6.39	brathl::CDoublePtrArray Class Reference	197
6.39.1	Detailed Description	198
6.40	brathl::CDoublePtrDoubleMap Class Reference	198
6.40.1	Detailed Description	199
6.41	brathl::CException Class Reference	199
6.41.1	Detailed Description	200
6.41.2	Constructor & Destructor Documentation	200
6.42	brathl::CExpressionException Class Reference	200
6.42.1	Detailed Description	201
6.42.2	Constructor & Destructor Documentation	202
6.43	brathl::CExpressionValue Class Reference	202
6.43.1	Detailed Description	203
6.44	brathl::CExternalFilesAvisoGrid Class Reference	203
6.44.1	Detailed Description	204
6.44.2	Member Function Documentation	205
6.45	brathl::CExternalFilesJason2 Class Reference	205
6.45.1	Detailed Description	205
6.46	brathl::CExternalFilesNetCDF Class Reference	205
6.46.1	Detailed Description	208
6.46.2	Member Function Documentation	208
6.47	brathl::CField Class Reference	208
6.47.1	Detailed Description	212
6.47.2	Member Data Documentation	213
6.48	brathl::CFieldArray Class Reference	213
6.48.1	Detailed Description	214
6.49	brathl::CFieldBasic Class Reference	215
6.49.1	Detailed Description	216
6.50	brathl::CFieldIndexData Class Reference	216
6.50.1	Detailed Description	218
6.51	brathl::CFieldNetCdf Class Reference	218

6.51.1	Detailed Description	222
6.51.2	Member Data Documentation	222
6.52	bratl::CFieldNetCdfCF Class Reference	223
6.52.1	Detailed Description	225
6.53	bratl::CFieldNetCdfCFAttr Class Reference	225
6.53.1	Detailed Description	227
6.54	bratl::CFieldRecord Class Reference	227
6.54.1	Detailed Description	229
6.55	bratl::CFieldSet Class Reference	229
6.55.1	Detailed Description	230
6.56	bratl::CFieldSetArrayDbf Class Reference	230
6.56.1	Detailed Description	232
6.57	bratl::CFieldSetDbf Class Reference	232
6.57.1	Detailed Description	234
6.58	bratl::CFieldSetString Class Reference	234
6.58.1	Detailed Description	236
6.59	bratl::CFile Class Reference	236
6.59.1	Detailed Description	237
6.59.2	Member Enumeration Documentation	238
6.59.3	Constructor & Destructor Documentation	238
6.59.4	Member Function Documentation	238
6.60	bratl::CFileException Class Reference	243
6.60.1	Detailed Description	244
6.60.2	Constructor & Destructor Documentation	244
6.61	bratl::CFileParams Class Reference	245
6.61.1	Detailed Description	246
6.61.2	Constructor & Destructor Documentation	246
6.61.3	Member Function Documentation	247
6.61.4	Member Data Documentation	247
6.62	bratl::CFloatArray Class Reference	248
6.62.1	Detailed Description	248
6.63	bratl::CProduct::CInfo Class Reference	248
6.63.1	Detailed Description	249
6.64	bratl::CInt16Array Class Reference	249
6.64.1	Detailed Description	250
6.65	bratl::CInt64Array Class Reference	250
6.65.1	Detailed Description	250
6.66	bratl::CInt8Array Class Reference	251
6.66.1	Detailed Description	251
6.67	bratl::CIntArray Class Reference	251

6.67.1 Detailed Description	252
6.68 bratl::CInternalFiles Class Reference	252
6.68.1 Detailed Description	254
6.69 bratl::CInternalFilesYFX Class Reference	254
6.69.1 Detailed Description	255
6.70 bratl::CInternalFilesZFX Class Reference	255
6.70.1 Detailed Description	256
6.71 bratl::CIntList Class Reference	257
6.71.1 Detailed Description	257
6.72 bratl::CIntMap Class Reference	257
6.72.1 Detailed Description	258
6.73 bratl::CField::CListField Class Reference	258
6.73.1 Detailed Description	259
6.73.2 Member Function Documentation	259
6.74 bratl::CProduct::CListInfo Class Reference	260
6.74.1 Detailed Description	260
6.75 bratl::CLoadAliasesException Class Reference	261
6.75.1 Detailed Description	261
6.75.2 Constructor & Destructor Documentation	262
6.76 bratl::CMapParameter Class Reference	262
6.76.1 Detailed Description	262
6.77 bratl::CMapProduct Class Reference	263
6.77.1 Detailed Description	264
6.78 bratl::CMemoryException Class Reference	264
6.78.1 Detailed Description	265
6.78.2 Constructor & Destructor Documentation	265
6.79 bratl::CMission Class Reference	265
6.79.1 Detailed Description	266
6.79.2 Constructor & Destructor Documentation	266
6.79.3 Member Function Documentation	267
6.79.4 Member Data Documentation	268
6.80 bratl::CObArray Class Reference	270
6.80.1 Detailed Description	270
6.81 bratl::CObDoubleMap Class Reference	271
6.81.1 Detailed Description	271
6.82 bratl::CObIntMap Class Reference	271
6.82.1 Detailed Description	272
6.83 bratl::CObList Class Reference	272
6.83.1 Detailed Description	273
6.84 bratl::CObMap Class Reference	274

6.84.1 Detailed Description	274
6.85 bratl::CObStack Class Reference	275
6.85.1 Detailed Description	275
6.86 bratl::CParameter Class Reference	275
6.86.1 Detailed Description	276
6.86.2 Constructor & Destructor Documentation	277
6.86.3 Member Function Documentation	277
6.87 bratl::CParameterException Class Reference	278
6.87.1 Detailed Description	279
6.87.2 Constructor & Destructor Documentation	279
6.88 CPlot Class Reference	279
6.88.1 Detailed Description	281
6.89 CPlotBase Class Reference	281
6.89.1 Detailed Description	282
6.90 CPlotField Class Reference	282
6.90.1 Detailed Description	283
6.91 bratl::CProductAop Class Reference	283
6.91.1 Detailed Description	284
6.91.2 Constructor & Destructor Documentation	284
6.92 bratl::CProductCryosat Class Reference	284
6.92.1 Detailed Description	285
6.92.2 Constructor & Destructor Documentation	285
6.93 bratl::CProductEnvisat Class Reference	285
6.93.1 Detailed Description	286
6.93.2 Constructor & Destructor Documentation	287
6.93.3 Member Function Documentation	287
6.94 bratl::CProductErs Class Reference	288
6.94.1 Detailed Description	289
6.94.2 Constructor & Destructor Documentation	289
6.94.3 Member Function Documentation	289
6.95 bratl::CProductErsWAP Class Reference	289
6.95.1 Detailed Description	291
6.95.2 Constructor & Destructor Documentation	291
6.95.3 Member Function Documentation	291
6.96 bratl::CProductException Class Reference	292
6.96.1 Detailed Description	292
6.96.2 Constructor & Destructor Documentation	293
6.97 bratl::CProductGfo Class Reference	293
6.97.1 Detailed Description	294
6.97.2 Constructor & Destructor Documentation	294

6.97.3 Member Function Documentation	294
6.98 brathl::CProductJason Class Reference	295
6.98.1 Detailed Description	295
6.98.2 Constructor & Destructor Documentation	296
6.98.3 Member Function Documentation	296
6.99 brathl::CProductJason2 Class Reference	296
6.99.1 Detailed Description	298
6.99.2 Constructor & Destructor Documentation	298
6.100brathl::CProductList Class Reference	298
6.100.1 Detailed Description	300
6.101brathl::CProductNetCdf Class Reference	300
6.101.1 Detailed Description	304
6.101.2 Constructor & Destructor Documentation	304
6.101.3 Member Data Documentation	304
6.102brathl::CProductNetCdfCF Class Reference	304
6.102.1 Detailed Description	306
6.102.2 Constructor & Destructor Documentation	306
6.102.3 Member Data Documentation	307
6.103brathl::CProductPodaac Class Reference	307
6.103.1 Detailed Description	308
6.103.2 Constructor & Destructor Documentation	308
6.104brathl::CProductRads Class Reference	308
6.104.1 Detailed Description	309
6.104.2 Constructor & Destructor Documentation	309
6.105brathl::CProductRiverLake Class Reference	309
6.105.1 Detailed Description	310
6.105.2 Constructor & Destructor Documentation	310
6.106brathl::CProductTopex Class Reference	310
6.106.1 Detailed Description	312
6.106.2 Constructor & Destructor Documentation	312
6.106.3 Member Function Documentation	312
6.106.4 Member Data Documentation	312
6.107brathl::CProductTopexSDR Class Reference	313
6.107.1 Detailed Description	314
6.107.2 Constructor & Destructor Documentation	314
6.107.3 Member Function Documentation	314
6.108brathl::CPtrMap Class Reference	315
6.108.1 Detailed Description	315
6.109brathl::CRecord Class Reference	315
6.109.1 Detailed Description	316

6.110brathl::CRecordSet Class Reference	317
6.110.1 Detailed Description	318
6.111brathl::CRegisteredPass Class Reference	318
6.111.1 Detailed Description	318
6.112brathl::CStringList Class Reference	319
6.112.1 Detailed Description	320
6.113brathl::CStringMap Class Reference	320
6.113.1 Detailed Description	320
6.114CTimeChangeEvent Class Reference	321
6.114.1 Detailed Description	321
6.114.2 Constructor & Destructor Documentation	321
6.114.3 Member Function Documentation	321
6.115CTimeChangeSpinButton Class Reference	322
6.115.1 Detailed Description	322
6.115.2 Constructor & Destructor Documentation	322
6.115.3 Member Function Documentation	322
6.116brathl::CTools Class Reference	323
6.116.1 Detailed Description	327
6.116.2 Member Function Documentation	327
6.117brathl::CTreeField Class Reference	347
6.117.1 Detailed Description	348
6.118brathl::CUInt16Array Class Reference	348
6.118.1 Detailed Description	349
6.119brathl::CUInt64Array Class Reference	349
6.119.1 Detailed Description	350
6.120brathl::CUInt8Array Class Reference	350
6.120.1 Detailed Description	351
6.121brathl::CUIntArray Class Reference	351
6.121.1 Detailed Description	352
6.122brathl::CUIntMap Class Reference	352
6.122.1 Detailed Description	353
6.123brathl::CUnImplementException Class Reference	353
6.123.1 Detailed Description	354
6.123.2 Constructor & Destructor Documentation	354
6.124CWPlot Class Reference	354
6.124.1 Detailed Description	355
6.125brathl::CXMLException Class Reference	356
6.125.1 Detailed Description	357
6.125.2 Constructor & Destructor Documentation	357
6.126brathl::CXMLParseException Class Reference	357

6.126.1 Detailed Description	358
6.126.2 Constructor & Destructor Documentation	358
6.127CZFXYPLOT Class Reference	359
6.127.1 Detailed Description	360
6.128vtkObArray Class Reference	360
6.128.1 Detailed Description	360
6.128.2 Constructor & Destructor Documentation	361
6.128.3 Member Function Documentation	361
6.129vtkObList Class Reference	361
6.129.1 Detailed Description	362
6.129.2 Constructor & Destructor Documentation	362
6.129.3 Member Function Documentation	362
6.130vtkObMap Class Reference	363
6.130.1 Detailed Description	363
6.130.2 Member Function Documentation	363
6.131wxObArray Class Reference	365
6.131.1 Detailed Description	365
6.131.2 Constructor & Destructor Documentation	365
6.131.3 Member Function Documentation	365
6.132wxObList Class Reference	366
6.132.1 Detailed Description	367
6.132.2 Constructor & Destructor Documentation	367
6.132.3 Member Function Documentation	367
6.133wxObMap Class Reference	367
6.133.1 Detailed Description	368
6.133.2 Member Function Documentation	368
7 File Documentation	369
7.1 brathl.h File Reference	369
7.1.1 Detailed Description	371
7.1.2 Macro Definition Documentation	371
7.1.3 Typedef Documentation	371
7.1.4 Enumeration Type Documentation	372
7.1.5 Variable Documentation	372
7.2 brathl_error.h File Reference	373
7.2.1 Detailed Description	374
7.3 brathlc.h File Reference	374
7.3.1 Detailed Description	376
7.3.2 Function Documentation	376
7.3.3 Variable Documentation	377

7.4	Exception.h File Reference	377
7.4.1	Detailed Description	378
7.5	MapParameter.h File Reference	378
7.5.1	Detailed Description	379

1 Module Index

1.1 Modules

Here is a list of all modules:

Error codes	16
Date error codes	18
Cycle/date conversion error codes	19
Algorithms classes	20
Tools	26
Criteria	66
Date conversion classes	83
Errors management	84
File services	85
Parameters	86
Date conversion C APIs	88
C API for reading data	95

2 Class Index

2.1 Class Hierarchy

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

_structDateDSM	97
_structDateJulian	97
_structDateSecond	98
_structDateYMDHMSM	99
bratl::CBratAlgoFilterGaussian1D	101
bratl::CBratAlgoFilterGaussian2D	102
bratl::CBratAlgoFilterLanczos1D	104
bratl::CBratAlgoFilterLanczos2D	106

brathl::CBratAlgoFilterLoess1D	107
brathl::CBratAlgoFilterLoess2D	111
brathl::CBratAlgoFilterMedian1D	114
brathl::CBratAlgoFilterMedian2D	117
brathl::CBratAlgorithmBase	120
brathl::CBratAlgorithmGeosVel	125
brathl::CBratAlgorithmGeosVelAtp	128
brathl::CBratAlgorithmGeosVelGrid	132
brathl::CBratAlgorithmGeosVelGridU	134
brathl::CBratAlgorithmGeosVelGridV	136
brathl::CCriteria	137
brathl::CCriteriaCycle	139
brathl::CCriteriaDatetime	145
brathl::CCriteriaLatLon	153
brathl::CCriteriaPass	160
brathl::CCriteriaPassInt	164
brathl::CCriteriaPassString	167
brathl::CCriterialInfo	152
brathl::CCriteriaCycleInfo	144
brathl::CCriteriaDatetimeInfo	150
brathl::CCriteriaLatLonInfo	159
brathl::CCriteriaPassInfo	162
brathl::CCriteriaPassIntInfo	166
brathl::CCriteriaPassStringInfo	170
brathl::CDate	173
brathl::CDatePeriod	190
brathl::CDoubleArray	195
brathl::CDoubleMap	196
brathl::CDoublePtrArray	197
brathl::CDoublePtrDoubleMap	198
brathl::CException	199
brathl::CAlgorithmException	99

brathl::CExpressionException	200
brathl::CFileException	243
brathl::CLoadAliasesException	261
brathl::CMemoryException	264
brathl::CParameterException	278
brathl::CProductException	292
brathl::CUnImplementException	353
brathl::CXMLException	356
brathl::CXMLParseException	357
brathl::CExpressionValue	202
brathl::CExternalFilesAvisoGrid	203
brathl::CExternalFilesJason2	205
brathl::CExternalFilesNetCDF	205
brathl::CField	208
brathl::CFieldArray	213
brathl::CFieldRecord	227
brathl::CFieldBasic	215
brathl::CFieldIndexData	216
brathl::CFieldNetCdf	218
brathl::CFieldNetCdfCF	223
brathl::CFieldNetCdfCFAttr	225
brathl::CFieldSet	229
brathl::CFieldSetArrayDbf	230
brathl::CFieldSetDbf	232
brathl::CFieldSetString	234
brathl::CFile	236
brathl::CFileParams	245
brathl::CFloatArray	248
brathl::CProduct::CInfo	248
brathl::CInt16Array	249
brathl::CInt64Array	250
brathl::CInt8Array	251

brathl::CIntArray	251
brathl::CInternalFiles	252
brathl::CInternalFilesYFX	254
brathl::CInternalFilesZFX	255
brathl::CIntList	257
brathl::CIntMap	257
brathl::CMapParameter	262
brathl::CMission	265
brathl::CObArray	270
brathl::CDataSet	171
brathl::CObDoubleMap	271
brathl::CObIntMap	271
brathl::CObList	272
brathl::CField::CListField	258
brathl::CProduct::CListInfo	260
brathl::CObMap	274
brathl::CMapProduct	263
brathl::CRecordSet	317
brathl::CObStack	275
brathl::CParameter	275
CPlotBase	281
CPlot	279
CWPlot	354
CZFXYPLOT	359
CPlotField	282
brathl::CProductAop	283
brathl::CProductCryosat	284
brathl::CProductEnvisat	285
brathl::CProductErs	288
brathl::CProductErsWAP	289
brathl::CProductGfo	293
brathl::CProductJason	295

brathl::CProductNetCdf	300
brathl::CProductNetCdfCF	304
brathl::CProductJason2	296
brathl::CProductPodaac	307
brathl::CProductRads	308
brathl::CProductRiverLake	309
brathl::CProductTopex	310
brathl::CProductTopexSDR	313
brathl::CPtrMap	315
brathl::CRecord	315
brathl::CRegisteredPass	318
brathl::CStringList	319
brathl::CProductList	298
brathl::CStringMap	320
CTimeChangeEvent	321
CTimeChangeSpinButton	322
brathl::CTools	323
brathl::CTreeField	347
brathl::CUInt16Array	348
brathl::CUInt64Array	349
brathl::CUInt8Array	350
brathl::CUIntArray	351
brathl::CUIntMap	352
vtkObArray	360
vtkObList	361
vtkObMap	363
wxObArray	365
wxObList	366
wxObMap	367

3 Class Index

3.1 Class List

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

_structDateDSM	97
_structDateJulian	97
_structDateSecond	98
_structDateYMDHMSM	99
brathl::CAlgorithmException	99
brathl::CBratAlgoFilterGaussian1D	101
brathl::CBratAlgoFilterGaussian2D	102
brathl::CBratAlgoFilterLanczos1D	104
brathl::CBratAlgoFilterLanczos2D	106
brathl::CBratAlgoFilterLoess1D	107
brathl::CBratAlgoFilterLoess2D	111
brathl::CBratAlgoFilterMedian1D	114
brathl::CBratAlgoFilterMedian2D	117
brathl::CBratAlgorithmBase	120
brathl::CBratAlgorithmGeosVel	125
brathl::CBratAlgorithmGeosVelAtp	128
brathl::CBratAlgorithmGeosVelGrid	132
brathl::CBratAlgorithmGeosVelGridU	134
brathl::CBratAlgorithmGeosVelGridV	136
brathl::CCriteria	137
brathl::CCriteriaCycle	139
brathl::CCriteriaCycleInfo	144
brathl::CCriteriaDatetime	145
brathl::CCriteriaDatetimeInfo	150
brathl::CCriterialInfo	152
brathl::CCriteriaLatLon	153
brathl::CCriteriaLatLonInfo	159
brathl::CCriteriaPass	160
brathl::CCriteriaPassInfo	162
brathl::CCriteriaPassInt	164

brathl::CCriteriaPassIntInfo	166
brathl::CCriteriaPassString	167
brathl::CCriteriaPassStringInfo	170
brathl::CDataSet	171
brathl::CDate	173
brathl::CDatePeriod	190
brathl::CDoubleArray	195
brathl::CDoubleMap	196
brathl::CDoublePtrArray	197
brathl::CDoublePtrDoubleMap	198
brathl::CException	199
brathl::CExpressionException	200
brathl::CExpressionValue	202
brathl::CExternalFilesAvisoGrid	203
brathl::CExternalFilesJason2	205
brathl::CExternalFilesNetCDF	205
brathl::CField	208
brathl::CFieldArray	213
brathl::CFieldBasic	215
brathl::CFieldIndexData	216
brathl::CFieldNetCdf	218
brathl::CFieldNetCdfCF	223
brathl::CFieldNetCdfCFAttr	225
brathl::CFieldRecord	227
brathl::CFieldSet	229
brathl::CFieldSetArrayDbI	230
brathl::CFieldSetDbI	232
brathl::CFieldSetString	234
brathl::CFile	236
brathl::CFileException	243
brathl::CFileParams	245
brathl::CFloatArray	248

brathl::CProduct::CInfo	248
brathl::CInt16Array	249
brathl::CInt64Array	250
brathl::CInt8Array	251
brathl::CIntArray	251
brathl::CInternalFiles	252
brathl::CInternalFilesYFX	254
brathl::CInternalFilesZFX	255
brathl::CIntList	257
brathl::CIntMap	257
brathl::CField::CListField	258
brathl::CProduct::CListInfo	260
brathl::CLoadAliasesException	261
brathl::CMapParameter	262
brathl::CMapProduct	263
brathl::CMemoryException	264
brathl::CMission	265
brathl::CObArray	270
brathl::CObDoubleMap	271
brathl::CObIntMap	271
brathl::CObList	272
brathl::CObMap	274
brathl::CObStack	275
brathl::CParameter	275
brathl::CParameterException	278
CPlot	279
CPlotBase	281
CPlotField	282
brathl::CProductAop	283
brathl::CProductCryosat	284
brathl::CProductEnvisat	285
brathl::CProductErs	288

brathl::CProductErsWAP	289
brathl::CProductException	292
brathl::CProductGfo	293
brathl::CProductJason	295
brathl::CProductJason2	296
brathl::CProductList	298
brathl::CProductNetCdf	300
brathl::CProductNetCdfCF	304
brathl::CProductPodaac	307
brathl::CProductRads	308
brathl::CProductRiverLake	309
brathl::CProductTopex	310
brathl::CProductTopexSDR	313
brathl::CPtrMap	315
brathl::CRecord	315
brathl::CRecordSet	317
brathl::CRegisteredPass	318
brathl::CStringList	319
brathl::CStringMap	320
CTimeChangeEvent	321
CTimeChangeSpinButton	322
brathl::CTools	323
brathl::CTreeField	347
brathl::CUInt16Array	348
brathl::CUInt64Array	349
brathl::CUInt8Array	350
brathl::CUIntArray	351
brathl::CUIntMap	352
brathl::CUnimplementException	353
CWPlot	354
brathl::CXMLException	356
brathl::CXMLParseException	357

CZFXYPLOT	359
vtkObArray	360
vtkObList	361
vtkObMap	363
wxObArray	365
wxObList	366
wxObMap	367

4 File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

AlgorithmDlg.h	??
Aliases.h	??
AliasesDictionary.h	??
AnimationToolbar.h	??
argtable2.h	??
AxisPropertyPanel.h	??
BitSet32.h	??
BratAlgoFilter.h	??
BratAlgoFilterGaussian.h	??
BratAlgoFilterGaussian1D.h	??
BratAlgoFilterGaussian2D.h	??
BratAlgoFilterKernel.h	??
BratAlgoFilterLanczos.h	??
BratAlgoFilterLanczos1D.h	??
BratAlgoFilterLanczos2D.h	??
BratAlgoFilterLoess.h	??
BratAlgoFilterLoess1D.h	??
BratAlgoFilterLoess2D.h	??
BratAlgoFilterMedian.h	??
BratAlgoFilterMedian1D.h	??
BratAlgoFilterMedian2D.h	??

BratAlgorithmBase.h	??
BratAlgorithmGeosVel.h	??
BratAlgorithmGeosVelAtp.h	??
BratAlgorithmGeosVelGrid.h	??
BratDisplay.h	??
BratDisplay_wdr.h	??
BratGui.h	??
BratGui_wdr.h	??
brathl.h	369
brathl_error.h	373
brathlc.h	374
BratLookupTable.h	??
BratObject.h	??
BratProcess.h	??
BratProcessExportAscii.h	??
BratProcessStats.h	??
BratProcessYFX.h	??
BratProcessZFX.h	??
BratTask.h	??
CallBack.h	??
CheckListBox.h	??
ColorPalette.h	??
ColorPicker.h	??
Config.h	??
ContourPropFrame.h	??
Criteria.h	??
CriteriaCycle.h	??
CriteriaDatetime.h	??
CriterialInfo.h	??
CriteriaLatLon.h	??
CriteriaPass.h	??
Dataset.h	??

display/DatasetPanel.h	??
gui/DatasetPanel.h	??
DatasetTreeCtrl.h	??
Date.h	??
DatePeriod.h	??
deelx.h	??
DelayDlg.h	??
Dictionary.h	??
DirTraverser.h	??
Display.h	??
DisplayDataTreeCtrl.h	??
DisplayPanel.h	??
Dnd.h	??
Exception.h	377
ExportDlg.h	??
Expression.h	??
ExternalFiles.h	??
ExternalFilesATP.h	??
ExternalFilesAvisoGrid.h	??
ExternalFilesFactory.h	??
ExternalFilesJason2.h	??
ExternalFilesNetCDF.h	??
Field.h	??
FieldsTreeCtrl.h	??
File.h	??
FileParams.h	??
Formula.h	??
FormulaDlg.h	??
Function.h	??
FunctionDlg.h	??
getopt.h	??
GuiFrame.h	??

GuiPanel.h	??
InternalFiles.h	??
InternalFilesFactory.h	??
InternalFilesYFX.h	??
InternalFilesZFX.h	??
LabeledTextCtrl.h	??
LatLonPoint.h	??
LatLonRect.h	??
List.h	??
LogPanel.h	??
LUTFrame.h	??
LUTPanel.h	??
MapColor.h	??
MapImageType.h	??
MapParameter.h	378
MapProjection.h	??
MapTypeDisp.h	??
Mission.h	??
NetCDFFiles.h	??
ObjectTree.h	??
Operation.h	??
OperationPanel.h	??
OperationTreeCtrl.h	??
Parameter.h	??
ParametersDictionary.h	??
Plot.h	??
PlotBase.h	??
PlotField.h	??
pragmalocation.h	??
Process.h	??
ProcessCommonTools.h	??
Product.h	??

ProductAop.h	??
ProductCryosat.h	??
ProductEnvisat.h	??
ProductErs.h	??
ProductErsWAP.h	??
ProductGfo.h	??
ProductJason.h	??
ProductJason2.h	??
ProductNetCdf.h	??
ProductNetCdfCF.h	??
ProductPodaac.h	??
ProductRads.h	??
ProductRiverLake.h	??
ProductTopex.h	??
ProductTopexSDR.h	??
ResolutionDlg.h	??
RichTextFrame.h	??
SchedulerTaskConfig.h	??
SchedulerTimer.h	??
SelectionCriteriaDlg.h	??
Stl.h	??
TaskList.h	??
TimeCtrl.h	??
Tools.h	??
Trace.h	??
TraceLog.h	??
TreeCtrl.h	??
TreeField.h	??
TreeWorkspace.h	??
TypedSaveFileDialog.h	??
Unit.h	??
Validators.h	??

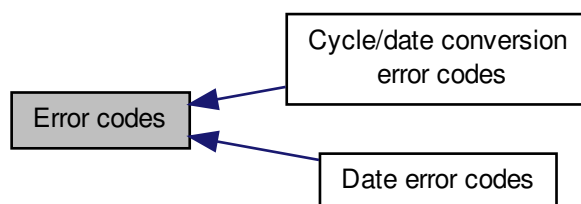
vtkBratArrowSource.h	??
vtkCameraState.h	??
VtkColor.h	??
vtkDataArrayPlotData.h	??
vtkGeoGridSource.h	??
vtkGeoMapFilter.h	??
vtkGSHHSReader.h	??
vtkInteractorStyle3DWPlot.h	??
vtkInteractorStyleWPlot.h	??
vtkInteractorStyleXYPlot.h	??
vtkInteractorStyleZFXYPLOT.h	??
vtkList.h	??
vtkNewAxisActor2D.h	??
vtkPlotData.h	??
vtkPlotDataCollection.h	??
vtkPointLocatorBrat.h	??
vtkProj2DFilter.h	??
vtkTools.h	??
vtkVelocityGlyphFilter.h	??
vtkXYPlotActor.h	??
vtkZFXYPLOTActor.h	??
vtkZFXYPLOTFilter.h	??
vtkZFXYPLOTFilterCollection.h	??
Win32MemLeaksAccurate.h	??
WindowHandler.h	??
Workspace.h	??
WorkspaceDlg.h	??
WorldPlotData.h	??
WorldPlotFrame.h	??
WorldPlotPanel.h	??
WPlot.h	??
WPlotPropertyPanel.h	??

<code>wxBatTools.h</code>	??
<code>wxDatePickerCtrl.h</code>	??
<code>wxList.h</code>	??
<code>wxVTKRenderWindowInteractor.h</code>	??
<code>Xml.h</code>	??
<code>XYPlotData.h</code>	??
<code>XYPlotFrame.h</code>	??
<code>XYPlotPanel.h</code>	??
<code>XYPlotPropertyPanel.h</code>	??
<code>ZFXYContourPropFrame.h</code>	??
<code>ZFXYPlot.h</code>	??
<code>ZFXYPlotData.h</code>	??
<code>ZFXYPlotFrame.h</code>	??
<code>ZFXYPlotPanel.h</code>	??
<code>ZFXYPlotPropertyPanel.h</code>	??

5 Module Documentation

5.1 Error codes

Collaboration diagram for Error codes:



Modules

- **Date error codes**
- **Cycle/date conversion error codes**

Macros

- **#define BRATHL_COUNT_ERROR -4**
Count error.
- **#define BRATHL_ERROR -1**
General error.
- **#define BRATHL_INCONSISTENCY_ERROR -11**
Inconsistency error.
- **#define BRATHL_IO_ERROR -7**
I/O error.
- **#define BRATHL_LIMIT_ERROR -6**
Limit error.
- **#define BRATHL_LOGIC_ERROR -10**
Logic error (program error)
- **#define BRATHL_MEMORY_ERROR -8**
Memory error.
- **#define BRATHL_RANGE_ERROR -5**
Range error.
- **#define BRATHL_SUCCESS 0**
- **#define BRATHL_SYNTAX_ERROR -2**
Syntax error.
- **#define BRATHL_SYSTEM_ERROR -9**
System error.
- **#define BRATHL_UNIMPLEMENT_ERROR -12**
error for non non implement code
- **#define BRATHL_UNIT_ERROR -3**
Unit error.
- **#define BRATHL_WARNING 2**
warning

5.1.1 Detailed Description

5.1.2 Macro Definition Documentation

5.1.2.1 #define BRATHL_SUCCESS 0

Success - no error

Referenced by `brathl::CDate::Add()`, `brathl::CDate::AddDays()`, `brathl_Cycle2YMDHMSM()`, `brathl_DayOfYear()`, `brathl_DiffDSM()`, `brathl_DiffJulian()`, `brathl_DiffYMDHMSM()`, `brathl_DSM2Julian()`, `brathl_DSM2Seconds()`, `brathl_DSM2YMDHMSM()`, `brathl_Errno2String()`, `brathl_Julian2DSM()`, `brathl_Julian2Seconds()`, `brathl_Julian2YMDHMSM()`, `brathl_NowYMDHMSM()`, `brathl_ReadData()`, `brathl_Seconds2DSM()`, `brathl_Seconds2Julian()`, `brathl_Seconds2YMDHMSM()`, `brathl_YMDHMSM2Cycle()`, `brathl_YMDHMSM2DSM()`, `brathl_YMDHMSM2Julian()`, `brathl_YMDHMSM2Seconds()`, `brathl::CDate::CheckDate()`, `brathl::CDate::CheckDay()`, `brathl::CDate::CheckHour()`, `brathl::CDate::CheckMinute()`, `brathl::CDate::CheckMonth()`, `brathl::CDate::CheckMuSecond()`, `brathl::CDate::CheckSecond()`, `brathl::CDate::CheckYear()`, `brathl::CMission::CMission()`, `brathl::CDate::ConstructDate()`, `brathl::CMission::Convert()`, `brathl::CDate::Convert2DecimalJulian()`, `brathl::CDate::Convert2DMM()`, `brathl::CDate::Convert2DSM()`, `brathl::CDate::Convert2Second()`, `brathl::CDate::Convert2SM()`, `brathl::CDate::Convert2YMDHMSM()`, `brathl::CMission::CtrlMission()`, `brathl::CDate::CvDate()`, `brathl::CDate::DayOfYear()`, `brathl::CDate::GetDay()`, `brathl::CDate::GetDaysInMonth()`, `brathl::CDate::GetHour()`, `brathl::CDate::GetMinute()`, `brathl::CDate::GetMonth()`, `brathl::CDate::GetMuSecond()`, `brathl::CDate::GetSecond()`, `brathl::CDate::GetYear()`, `brathl::CMission::LoadAliasName()`, `brathl::CDate::SetDate()`, `brathl::CDate::SetDateJulian()`, `brathl::CDate::SetDateNow()`, `brathl::CDatePeriod::SetFrom()`, `brathl::CDatePeriod::SetTo()`, and `brathl::CDate::SubtractDays()`.

5.2 Date error codes

Collaboration diagram for Date error codes:



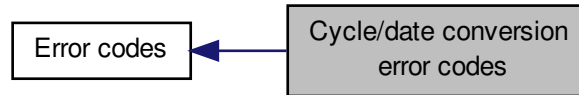
Macros

- #define **BRATHL_ERROR_INVALID_DATE** -101
Invalid date.
- #define **BRATHL_ERROR_INVALID_DATE_NEGATIVE** -112
Invalid date (date must be > 0)
- #define **BRATHL_ERROR_INVALID_DATE_REF** -102
Invalid reference date.
- #define **BRATHL_ERROR_INVALID_DATE_REF_CONV** -103
Invalid reference date conversion.
- #define **BRATHL_ERROR_INVALID_DAY** -107
Invalid day value.
- #define **BRATHL_ERROR_INVALID_DSM** -104
Invalid days or seconds or museonds values (must be > 0)
- #define **BRATHL_ERROR_INVALID_HOUR** -108
Invalid hour value (must be >= 0 and <= 23)
- #define **BRATHL_ERROR_INVALID_MINUTE** -109
Invalid minute value (must be >= 0 and <= 59)
- #define **BRATHL_ERROR_INVALID_MONTH** -106
Invalid month value (must be >= 1 and <= 12)
- #define **BRATHL_ERROR_INVALID_MUSECOND** -111
Invalid musecond value (must be >= 0 and <= 999999)
- #define **BRATHL_ERROR_INVALID_SECOND** -110
Invalid second value (must be >= 0 and <= 59)
- #define **BRATHL_ERROR_INVALID_YEAR** -105
Invalid year value (must be >= 1950)

5.2.1 Detailed Description

5.3 Cycle/date conversion error codes

Collaboration diagram for Cycle/date conversion error codes:



Macros

- #define **BRATHL_ERROR_INVALID_MISSION** -203
Unknown mission value.
- #define **BRATHL_ERROR_INVALID_NB_PASS** -201
Invalid nb pass (must be > 0)
- #define **BRATHL_ERROR_INVALID_REPETITION** -202
Invalid repetition (must be > 0)
- #define **BRATHL_WARNING_INVALID_REF_FILE_FIELD** -205
WARNING - Invalid reference mission file format.
- #define **BRATHL_WARNING_INVALID_REF_FILE_FIELDDATE** -206
WARNING - Invalid reference mission date.
- #define **BRATHL_WARNING_OPEN_FILE_ALIAS_MISSION** -207
WARNING - Unable to open alias mission file.
- #define **BRATHL_WARNING_OPEN_FILE_REF_FILE** -204
WARNING - Unable to open reference mission file.

5.3.1 Detailed Description

5.4 Algorithms classes

Classes

- class **bratl::CBratAlgoFilterGaussian1D**
- class **bratl::CBratAlgoFilterGaussian2D**
- class **bratl::CBratAlgoFilterLanczos1D**
- class **bratl::CBratAlgoFilterLanczos2D**
- class **bratl::CBratAlgoFilterLoess1D**
- class **bratl::CBratAlgoFilterLoess2D**
- class **bratl::CBratAlgoFilterMedian1D**
- class **bratl::CBratAlgoFilterMedian2D**
- class **bratl::CBratAlgorithmBase**
- class **bratl::CBratAlgorithmGeosVel**
- class **bratl::CBratAlgorithmGeosVelAtp**
- class **bratl::CBratAlgorithmGeosVelGrid**
- class **bratl::CBratAlgorithmGeosVelGridU**
- class **bratl::CBratAlgorithmGeosVelGridV**

Macros

- **#define AUTO_REGISTER_BASE(base) CBratAlgorithmBaseRegistration _base_registration_ ## base(&base_factory<base>);**

Typedefs

- typedef **CBratAlgorithmBase** *(* **bratl::base_creator**)(void)
- typedef map< string, **CBratAlgorithmBase** * > **bratl::mapbratalgorithmbase**
- typedef vector< **CBratAlgorithmBase** * > **bratl::vectorbratalgorithmbase**

Functions

- template<class T > **CBratAlgorithmBase** * **bratl::base_factory** ()
- **bratl::CBratAlgorithmGeosVelGrid::CBratAlgorithmGeosVelGrid** ()
- **bratl::CBratAlgorithmGeosVelGrid::CBratAlgorithmGeosVelGrid** (const **CBratAlgorithmGeosVelGrid** ©)
- **bratl::CBratAlgorithmGeosVelGridU::CBratAlgorithmGeosVelGridU** ()
- **bratl::CBratAlgorithmGeosVelGridU::CBratAlgorithmGeosVelGridU** (const **CBratAlgorithmGeosVelGridU** ©)
- **bratl::CBratAlgorithmGeosVelGridV::CBratAlgorithmGeosVelGridV** ()
- **bratl::CBratAlgorithmGeosVelGridV::CBratAlgorithmGeosVelGridV** (const **CBratAlgorithmGeosVelGridV** ©)
- void **bratl::CBratAlgorithmGeosVelGrid::CheckEquatorLimit** ()
- virtual void **bratl::CBratAlgorithmGeosVelGrid::CheckInputParams** (CVectorBratAlgorithmParam &args)
- void **bratl::CBratAlgorithmGeosVelGrid::CheckLatLonExpression** (uint32_t index)
- void **bratl::CBratAlgorithmGeosVelGrid::CheckProduct** ()
- void **bratl::CBratAlgorithmGeosVelGrid::CheckVarExpression** (uint32_t index)
- double **bratl::CBratAlgorithmGeosVelGrid::ComputeMean** ()
- double **bratl::CBratAlgorithmGeosVelGrid::ComputeSingle** ()
- virtual double **bratl::CBratAlgorithmGeosVelGrid::ComputeVelocity** ()=0
- double **bratl::CBratAlgorithmGeosVelGridU::ComputeVelocity** ()

- double **bratl::CBratAlgorithmGeosVelGridV::ComputeVelocity** ()
- virtual void **bratl::CBratAlgorithmGeosVelGrid::DeleteFieldNetCdf** ()
- virtual void **bratl::CBratAlgorithmGeosVelGrid::DeleteProduct** ()
- virtual void **bratl::CBratAlgorithmGeosVelGrid::Dump** (ostream &fOut=cerr)
- virtual void **bratl::CBratAlgorithmGeosVelGridU::Dump** (ostream &fOut=cerr)
- virtual void **bratl::CBratAlgorithmGeosVelGridV::Dump** (ostream &fOut=cerr)
- virtual string **bratl::CBratAlgorithmGeosVelGridU::GetDescription** ()
- virtual string **bratl::CBratAlgorithmGeosVelGridV::GetDescription** ()
- virtual string **bratl::CBratAlgorithmGeosVelGrid::GetInputParamDesc** (uint32_t indexParam)
- virtual
 CBratAlgorithmParam::bratAlgoParamTypeVal **bratl::CBratAlgorithmGeosVelGrid::GetInputParam-**
 Format (uint32_t indexParam)
- virtual string **bratl::CBratAlgorithmGeosVelGrid::GetInputParamUnit** (uint32_t indexParam)
- uint32_t **bratl::CBratAlgorithmGeosVelGrid::GetLatDimRange** (CFieldNetCdf *field)
- int32_t **bratl::CBratAlgorithmGeosVelGrid::GetLatitudeIndex** (double value)
- void **bratl::CBratAlgorithmGeosVelGrid::GetLatitudes** ()
- uint32_t **bratl::CBratAlgorithmGeosVelGrid::GetLonDimRange** (CFieldNetCdf *field)
- int32_t **bratl::CBratAlgorithmGeosVelGrid::GetLongitudeIndex** (double value)
- void **bratl::CBratAlgorithmGeosVelGrid::GetLongitudes** ()
- virtual string **bratl::CBratAlgorithmGeosVelGridU::GetName** ()
- virtual string **bratl::CBratAlgorithmGeosVelGridV::GetName** ()
- virtual uint32_t **bratl::CBratAlgorithmGeosVelGrid::GetNumInputParam** ()
- virtual string **bratl::CBratAlgorithmGeosVelGrid::GetOutputUnit** ()
- virtual double **bratl::CBratAlgorithmGeosVelGrid::GetParamDefaultValue** (uint32_t indexParam)
- virtual string **bratl::CBratAlgorithmGeosVelGrid::GetParamName** (uint32_t indexParam)
- void **bratl::CBratAlgorithmGeosVelGrid::GetVarCacheExpressionValue** (int32_t minIndexLat, int32_t -
 t maxIndexLat, int32_t minIndexLon, int32_t maxIndexLon)
- double **bratl::CBratAlgorithmGeosVelGrid::GetVarExpressionValue** (int32_t indexLat, int32_t indexLon)
- double **bratl::CBratAlgorithmGeosVelGrid::GetVarExpressionValueCache** (int32_t indexLat, int32_t
 indexLon)
- void **bratl::CBratAlgorithmGeosVelGrid::Init** ()
- void **bratl::CBratAlgorithmGeosVelGridU::Init** ()
- void **bratl::CBratAlgorithmGeosVelGridV::Init** ()
- virtual void **bratl::CBratAlgorithmGeosVelGrid::OpenProductFile** ()
- **CBratAlgorithmGeosVelGrid & bratl::CBratAlgorithmGeosVelGrid::operator=** (const CBratAlgorithm-
 GeosVelGrid ©)
- bool **bratl::CBratAlgorithmGeosVelGrid::PrepareComputeVelocity** ()
- virtual void **bratl::CBratAlgorithmGeosVelGrid::PrepareDataReading2D** (int32_t minIndexLat, int32_t -
 t maxIndexLat, int32_t minIndexLon, int32_t maxIndexLon)
- virtual void **bratl::CBratAlgorithmGeosVelGrid::PrepareDataReading2D** (int32_t indexLat, int32_t index-
 Lon)
- virtual void **bratl::CBratAlgorithmGeosVelGrid::PrepareDataValues2DComplexExpression** (C-
 ExpressionValue &exprValue)
- virtual void **bratl::CBratAlgorithmGeosVelGrid::PrepareDataValues2DComplexExpressionWithAlgo**
 (**CExpressionValue** &exprValue)
- virtual void **bratl::CBratAlgorithmGeosVelGrid::PrepareDataValues2DOneField** (CExpressionValue
 &exprValue)
- virtual double **bratl::CBratAlgorithmGeosVelGrid::Run** (CVectorBratAlgorithmParam &args)
- void **bratl::CBratAlgorithmGeosVelGrid::Set** (const CBratAlgorithmGeosVelGrid ©)
- void **bratl::CBratAlgorithmGeosVelGrid::SetBeginOfFile** ()
- void **bratl::CBratAlgorithmGeosVelGrid::SetEndOfFile** ()
- virtual void **bratl::CBratAlgorithmGeosVelGrid::SetParamValues** (CVectorBratAlgorithmParam &args)
- virtual **bratl::CBratAlgorithmGeosVelGrid::~CBratAlgorithmGeosVelGrid** ()
- virtual **bratl::CBratAlgorithmGeosVelGridU::~CBratAlgorithmGeosVelGridU** ()
- virtual **bratl::CBratAlgorithmGeosVelGridV::~CBratAlgorithmGeosVelGridV** ()

Variables

- bool **bratl::CBratAlgorithmGeosVelGrid::m_allLongitudes**
- static const uint32_t **bratl::CBratAlgorithmGeosVelGrid::m_EQUATOR_LAT_LIMIT_INDEX = 3**
- double **bratl::CBratAlgorithmGeosVelGrid::m_equatorLimit**
- CFieldNetCdf * **bratl::CBratAlgorithmGeosVelGrid::m_fieldLat**
- CFieldNetCdf * **bratl::CBratAlgorithmGeosVelGrid::m_fieldLon**
- int32_t **bratl::CBratAlgorithmGeosVelGrid::m_indexLat**
- int32_t **bratl::CBratAlgorithmGeosVelGrid::m_indexLon**
- static const uint32_t **bratl::CBratAlgorithmGeosVelGrid::m_INPUT_PARAMS = 4**
- static const uint32_t **bratl::CBratAlgorithmGeosVelGrid::m_LAT_PARAM_INDEX = 0**
- CDoubleArray **bratl::CBratAlgorithmGeosVelGrid::m_latitudes**
- static const uint32_t **bratl::CBratAlgorithmGeosVelGrid::m_LON_PARAM_INDEX = 1**
- CDoubleArray **bratl::CBratAlgorithmGeosVelGrid::m_longitudes**
- double **bratl::CBratAlgorithmGeosVelGrid::m_lonMax**
- double **bratl::CBratAlgorithmGeosVelGrid::m_lonMin**
- CExpressionValue **bratl::CBratAlgorithmGeosVelGrid::m_rawDataCache**
- static const uint32_t **bratl::CBratAlgorithmGeosVelGrid::m_VAR_PARAM_INDEX = 2**
- int32_t **bratl::CBratAlgorithmGeosVelGrid::m_varDimLatIndex**
- int32_t **bratl::CBratAlgorithmGeosVelGrid::m_varDimLonIndex**
- double **bratl::CBratAlgorithmGeosVelGrid::m_varValue**
- double **bratl::CBratAlgorithmGeosVelGrid::m_varValueE**
- double **bratl::CBratAlgorithmGeosVelGrid::m_varValueN**
- double **bratl::CBratAlgorithmGeosVelGrid::m_varValueS**
- double **bratl::CBratAlgorithmGeosVelGrid::m_varValueW**

5.4.1 Detailed Description

5.4.2 Function Documentation

5.4.2.1 **bratl::CBratAlgorithmGeosVelGrid::CBratAlgorithmGeosVelGrid ()**

Default constructor

5.4.2.2 **bratl::CBratAlgorithmGeosVelGrid::CBratAlgorithmGeosVelGrid (const CBratAlgorithmGeosVelGrid & copy)**

Copy constructor

5.4.2.3 **bratl::CBratAlgorithmGeosVelGridU::CBratAlgorithmGeosVelGridU ()**

Default constructor

5.4.2.4 **bratl::CBratAlgorithmGeosVelGridU::CBratAlgorithmGeosVelGridU (const CBratAlgorithmGeosVelGridU & copy)**

Copy constructor

5.4.2.5 **bratl::CBratAlgorithmGeosVelGridV::CBratAlgorithmGeosVelGridV ()**

Default constructor

5.4.2.6 **bratl::CBratAlgorithmGeosVelGridV::CBratAlgorithmGeosVelGridV (const CBratAlgorithmGeosVelGridV & copy)**

Copy constructor

5.4.2.7 `void bratl::CBratAlgorithmGeosVelGrid::Dump (ostream & fOut = cerr) [virtual]`

Dump function

Reimplemented from **bratl::CBratAlgorithmGeosVel** (p. 128).

Reimplemented in **bratl::CBratAlgorithmGeosVelGridV** (p. 23), and **bratl::CBratAlgorithmGeosVelGridU** (p. 23).

References **bratl::CBratAlgorithmGeosVel::Dump()**.

Referenced by **bratl::CBratAlgorithmGeosVelGridU::Dump()**, and **bratl::CBratAlgorithmGeosVelGridV::Dump()**.

5.4.2.8 `void bratl::CBratAlgorithmGeosVelGridU::Dump (ostream & fOut = cerr) [virtual]`

Dump function

Reimplemented from **bratl::CBratAlgorithmGeosVelGrid** (p. 23).

References **bratl::CBratAlgorithmGeosVelGrid::Dump()**.

5.4.2.9 `void bratl::CBratAlgorithmGeosVelGridV::Dump (ostream & fOut = cerr) [virtual]`

Dump function

Reimplemented from **bratl::CBratAlgorithmGeosVelGrid** (p. 23).

References **bratl::CBratAlgorithmGeosVelGrid::Dump()**.

5.4.2.10 `virtual string bratl::CBratAlgorithmGeosVelGridU::GetDescription () [inline],[virtual]`

Gets the description of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

5.4.2.11 `virtual string bratl::CBratAlgorithmGeosVelGridV::GetDescription () [inline],[virtual]`

Gets the description of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

5.4.2.12 `virtual string bratl::CBratAlgorithmGeosVelGrid::GetInputParamDesc (uint32_t indexParam) [inline],[virtual]`

Gets the description of an input parameter.

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References **bratl::CTools::Format()**.

5.4.2.13 `virtual CBratAlgorithmParam::bratAlgoParamTypeVal bratl::CBratAlgorithmGeosVelGrid::GetInputParamFormat (uint32_t indexParam) [inline],[virtual]`

Gets the format of an input parameter : **CBratAlgorithmParam::T_DOUBLE** for double **CBratAlgorithmParam::T_FLOAT** for float **CBratAlgorithmParam::T_INT** for integer **CBratAlgorithmParam::T_LONG** for long integer **CBratAlgorithmParam::T_STRING** for string **CBratAlgorithmParam::T_CHAR** for a character

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References `bratl::CTools::Format()`.

5.4.2.14 `virtual string bratl::CBratAlgorithmGeosVelGrid::GetInputParamUnit (uint32_t indexParam) [inline], [virtual]`

Gets the unit of an input parameter :

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **`bratl::CBratAlgorithmBase`** (p. 124).

References `bratl::CTools::Format()`.

5.4.2.15 `virtual string bratl::CBratAlgorithmGeosVelGridU::GetName () [inline], [virtual]`

Gets the name of the algorithm

Implements **`bratl::CBratAlgorithmBase`** (p. 124).

5.4.2.16 `virtual string bratl::CBratAlgorithmGeosVelGridV::GetName () [inline], [virtual]`

Gets the name of the algorithm

Implements **`bratl::CBratAlgorithmBase`** (p. 124).

5.4.2.17 `virtual uint32_t bratl::CBratAlgorithmGeosVelGrid::GetNumInputParam () [inline], [virtual]`

Gets the number of input parameters to pass to the 'Run' function

Implements **`bratl::CBratAlgorithmBase`** (p. 125).

5.4.2.18 `virtual string bratl::CBratAlgorithmGeosVelGrid::GetOutputUnit () [inline], [virtual]`

Gets the unit of an output value returned by the 'Run' function.

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **`bratl::CBratAlgorithmBase`** (p. 125).

5.4.2.19 `CBratAlgorithmGeosVelGrid & bratl::CBratAlgorithmGeosVelGrid::operator= (const CBratAlgorithmGeosVelGrid & copy)`

Overloads operator '='

5.4.2.20 `double bratl::CBratAlgorithmGeosVelGrid::Run (CVectorBratAlgorithmParam & args) [virtual]`

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

Returns

the result of the execution

Implements **`bratl::CBratAlgorithmBase`** (p. 125).

5.4.2.21 `bratl::CBratAlgorithmGeosVelGrid::~~CBratAlgorithmGeosVelGrid () [virtual]`

Destructor

5.4.2.22 `bratl::CBratAlgorithmGeosVelGridU::~~CBratAlgorithmGeosVelGridU () [virtual]`

Destructor

5.4.2.23 `bratl::CBratAlgorithmGeosVelGridV::~~CBratAlgorithmGeosVelGridV () [virtual]`

Destructor

5.5 Tools

Classes

- class **brathl::CDoubleArray**
- class **brathl::CDoubleMap**
- class **brathl::CDoublePtrArray**
- class **brathl::CDoublePtrDoubleMap**
- class **brathl::CExpressionValue**
- class **brathl::CExternalFilesAvisoGrid**
- class **brathl::CExternalFilesJason2**
- class **brathl::CExternalFilesNetCDF**
- class **brathl::CFloatArray**
- class **brathl::CInt16Array**
- class **brathl::CInt64Array**
- class **brathl::CInt8Array**
- class **brathl::CIntArray**
- class **brathl::CInternalFiles**
- class **brathl::CInternalFilesYFX**
- class **brathl::CInternalFilesZFX**
- class **brathl::CIntList**
- class **brathl::CIntMap**
- class **brathl::CObArray**
- class **brathl::CObDoubleMap**
- class **brathl::CObIntMap**
- class **brathl::CObList**
- class **brathl::CObMap**
- class **brathl::CObStack**
- class **brathl::CPtrMap**
- class **brathl::CRegisteredPass**
- class **brathl::CStringList**
- class **brathl::CStringMap**
- class **brathl::CTools**
- class **brathl::CUInt16Array**
- class **brathl::CUInt64Array**
- class **brathl::CUInt8Array**
- class **brathl::CUIntArray**
- class **brathl::CUIntMap**

Macros

- #define **ADD_OFFSET_ATTR** "add_offset"
- #define **AT_BEGINNING** 0xFFFFFFFFUL
- #define **AXIS_ATTR** "axis"
- #define **COMMENT_ATTR** "comment"
- #define **CONVENTIONS_ATTR** "Conventions"
- #define **DATA_SET_ATTR** "data_set"
- #define **FILE_TITLE_ATTR** "title"
- #define **FILE_TYPE_ATTR** "FileType"
- #define **FILL_VALUE_ATTR** "_FillValue"
- #define **LONG_NAME_ATTR** "long_name"
- #define **MISSION_NAME_ATTR** "mission_name"
- #define **PRODUCT_TYPE_ATTR** "product_type"
- #define **SCALE_FACTOR_ATTR** "scale_factor"

- `#define STANDARD_NAME_ATTR "standard_name"`
- `#define TITLE_ATTR "title"`
- `#define UNITS_ATTR "units"`
- `#define VALID_MAX_ATTR "valid_max"`
- `#define VALID_MIN_ATTR "valid_min"`

Typedefs

- `typedef vector< doublearray > brathl::arraydoublearray`
- `typedef vector< doubleptrarray > brathl::arraydoubleptrarray`
- `typedef map< string, CStringArray > brathl::maparraystring`
- `typedef map< string, CObjectTreeNode * > brathl::mapTreeNode`
- `typedef vector< CObjectTreeNode * > brathl::vectorTreeNode`

Functions

- `void brathl::CArrayDoublePtrArray::AdjustValidMinMax (double value)`
- `void brathl::CArrayDoubleArray::AdjustValidMinMax (double value)`
- `DoublePtr brathl::CMatrix::At (uint32_t i, uint32_t j)`
- `CExternalFiles * brathl::BuildExistingExternalFileKind (const string &Name)`
- `CInternalFiles * brathl::BuildExistingInternalFileKind (const string &name, const CStringArray *fieldNames)`
- `brathl::CArrayDoubleArray::CArrayDoubleArray ()`
Empty CDoubleArray (p. 195) ctor.
- `brathl::CArrayDoubleArray::CArrayDoubleArray (const CArrayDoubleArray &a)`
- `brathl::CArrayDoublePtrArray::CArrayDoublePtrArray (bool bDelete=true)`
Empty CDoubleArray (p. 195) ctor.
- `brathl::CArrayDoublePtrArray::CArrayDoublePtrArray (const CArrayDoublePtrArray &a)`
- `brathl::CArrayStringMap::CArrayStringMap ()`
CStringMap (p. 320) ctor.
- `brathl::CArrayStringMap::CArrayStringMap (const CArrayStringMap &a)`
- `brathl::CDoubleArray::CDoubleArray ()`
Empty CDoubleArray (p. 195) ctor.
- `brathl::CDoubleArray::CDoubleArray (const CDoubleArray &vect)`
- `brathl::CDoubleArrayOb::CDoubleArrayOb (const CDoubleArrayOb &vect)`
- `brathl::CDoubleMap::CDoubleMap ()`
CDoubleMap (p. 196) ctor.
- `brathl::CDoublePtrArray::CDoublePtrArray (bool bDelete=true)`
Empty CDoublePtrArray (p. 197) ctor.
- `brathl::CDoublePtrDoubleMap::CDoublePtrDoubleMap (bool bDelete=true)`
CDoublePtrDoubleMap (p. 198) ctor.
- `brathl::CDoublePtrDoubleMap::CDoublePtrDoubleMap (const CUIntArray &matrixDims, bool bDelete=true)`
- `brathl::CFloatArray::CFloatArray ()`
Empty CFloatArray (p. 248) ctor.
- `brathl::CFloatArray::CFloatArray (const CFloatArray &vect)`
- `brathl::CInt16Array::CInt16Array ()`
Empty CInt16Array (p. 249) ctor.
- `brathl::CInt16Array::CInt16Array (const CInt16Array &vect)`
- `brathl::CInt64Array::CInt64Array ()`
Empty CInt64Array (p. 250) ctor.

- **bratl::CInt64Array::CInt64Array** (const **CInt64Array** &v)
- **bratl::CInt8Array::CInt8Array** ()
Empty CInt8Array (p. 251) ctor.
- **bratl::CInt8Array::CInt8Array** (const **CInt8Array** &vect)
- **bratl::CIntArray::CIntArray** ()
Empty CIntArray (p. 251) ctor.
- **bratl::CIntArray::CIntArray** (const **CIntArray** &vect)
- **bratl::CIntList::CIntList** ()
Empty CIntList (p. 257) ctor.
- **bratl::CIntList::CIntList** (const **CIntList** &list)
- **bratl::CIntMap::CIntMap** ()
CIntMap (p. 257) ctor.
- virtual CBratObject * **bratl::CDoubleArrayOb::Clone** ()
- virtual CBratObject * **bratl::CObArrayOb::Clone** ()
- **bratl::CMatrix::CMatrix** (const CMatrix &m)
- **bratl::CMatrixDouble::CMatrixDouble** (uint32_t nrows, uint32_t ncols)
- **bratl::CMatrixDouble::CMatrixDouble** (const CMatrixDouble &m)
- **bratl::CMatrixDoublePtr::CMatrixDoublePtr** (uint32_t nrows, uint32_t ncols)
- **bratl::CMatrixDoublePtr::CMatrixDoublePtr** (const CMatrixDoublePtr &m)
- **bratl::CObArray::CObArray** (bool bDelete=true)
Empty CObArray (p. 270) ctor.
- **bratl::CObArray::CObArray** (const **CObArray** &vect)
- **bratl::CObArrayOb::CObArrayOb** (bool bDelete=true)
- **bratl::CObArrayOb::CObArrayOb** (const CObArrayOb &vect)
- **bratl::CObDoubleMap::CObDoubleMap** (bool bDelete=true)
CObMap (p. 274) ctor.
- **bratl::CObIntMap::CObIntMap** (bool bDelete=true)
CObMap (p. 274) ctor.
- **bratl::CObList::CObList** (bool bDelete=true)
Empty CObList (p. 272) ctor.
- **bratl::CObList::CObList** (const **CObList** &lst)
- **bratl::CObMap::CObMap** (bool bDelete=true)
CObMap (p. 274) ctor.
- **bratl::CObMap::CObMap** (const **CObMap** &obMap)
- **bratl::CObStack::CObStack** (bool bDelete=true)
Empty CObArray (p. 270) ctor.
- virtual bool **bratl::CStringList::Complement** (const **CStringList** &array, **CStringList** &complement) const
- virtual bool **bratl::CStringArray::Complement** (const CStringArray &array, CStringArray &complement) const
- virtual bool **bratl::CUIIntArray::Complement** (const **CUIIntArray** &array, **CUIIntArray** &complement) const
- virtual bool **bratl::CUIInt16Array::Complement** (const **CUIInt16Array** &array, **CUIInt16Array** &complement) const
- virtual bool **bratl::CUIInt8Array::Complement** (const **CUIInt8Array** &array, **CUIInt8Array** &complement) const
- **bratl::CPtrMap::CPtrMap** (bool bDelete=true)
CPtrMap (p. 315) ctor.
- **bratl::CStringArray::CStringArray** ()
Empty CStringArray ctor.
- **bratl::CStringArray::CStringArray** (const CStringArray &vect)
- **bratl::CStringArray::CStringArray** (const stringarray &vect)
- **bratl::CStringArray::CStringArray** (const **CStringList** &lst)
- **bratl::CStringArray::CStringArray** (const stringlist &lst)

- **brathl::CStringList::CStringList ()**
Empty CStringList (p. 319) ctor.
- **brathl::CStringList::CStringList (const CStringList &list)**
- **brathl::CStringList::CStringList (const stringlist &list)**
- **brathl::CStringList::CStringList (const CStringArray &vect)**
- **brathl::CStringList::CStringList (const stringarray &vect)**
- **brathl::CStringMap::CStringMap ()**
CStringMap (p. 320) ctor.
- **brathl::CUInt16Array::CUInt16Array ()**
Empty CUInt16Array (p. 348) ctor.
- **brathl::CUInt16Array::CUInt16Array (const CUInt16Array &vect)**
- **brathl::CUInt64Array::CUInt64Array ()**
Empty CUIntArray (p. 351) ctor.
- **brathl::CUInt64Array::CUInt64Array (const CUInt64Array &vect)**
- **brathl::CUInt8Array::CUInt8Array ()**
Empty CUInt8Array (p. 350) ctor.
- **brathl::CUInt8Array::CUInt8Array (const CUInt8Array &vect)**
- **brathl::CUIntArray::CUIntArray ()**
Empty CUIntArray (p. 351) ctor.
- **brathl::CUIntArray::CUIntArray (const CUIntArray &vect)**
- **brathl::CUIntMap::CUIntMap ()**
CUIntMap (p. 352) ctor.
- **const double * brathl::CDoubleArray::data () const**
- **void brathl::CDoublePtrArray::Delete (DoublePtr matrix)**
- **void brathl::CArrayDoublePtrArray::Delete (DoublePtr matrix)**
- **void brathl::CDoublePtrDoubleMap::Delete (DoublePtr *matrix)**
- **virtual void brathl::CStringList::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CIntList::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CObList::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CStringArray::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CIntArray::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CInt64Array::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CUIntArray::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CUInt64Array::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CInt16Array::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CUInt16Array::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CInt8Array::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CUInt8Array::Dump (ostream &fOut=cerr) const**
Dump fonction.
- **virtual void brathl::CFloatArray::Dump (ostream &fOut=cerr) const**

- Dump fonction.*
- virtual void **brathl::CDoubleArray::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CDoublePtrArray::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CArrayDoublePtrArray::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CArrayDoubleArray::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CArrayStringMap::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CDoubleArrayOb::Dump** (ostream &fOut=cerr) const
- virtual void **brathl::CObArray::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CObArrayOb::Dump** (ostream &fOut=cerr) const
- virtual void **brathl::CStringMap::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CIntMap::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CUIntMap::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CDoubleMap::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CObMap::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CObIntMap::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CObDoubleMap::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CDoublePtrDoubleMap::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CPtrMap::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CMatrix::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CMatrixDoublePtr::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CMatrixDouble::Dump** (ostream &fOut=cerr) const
- Dump fonction.*
- virtual void **brathl::CStringList::Erase** (const string &str)
- virtual void **brathl::CStringList::Erase** (CStringList::iterator it)
- bool **brathl::CObList::Erase** (CBratObject *ob)
- virtual bool **brathl::CObList::Erase** (CObList::iterator it)
- virtual bool **brathl::CStringArray::Erase** (CStringArray::iterator it)
- virtual bool **brathl::CStringArray::Erase** (int32_t index)
- virtual bool **brathl::CStringArray::Erase** (uint32_t index)
- virtual bool **brathl::CIntArray::Erase** (CIntArray::iterator it)
- virtual bool **brathl::CInt64Array::Erase** (CInt64Array::iterator it)
- virtual bool **brathl::CUIntArray::Erase** (CUIntArray::iterator it)
- virtual bool **brathl::CUInt64Array::Erase** (CUInt64Array::iterator it)
- virtual bool **brathl::CInt16Array::Erase** (CInt16Array::iterator it)
- virtual bool **brathl::CUInt16Array::Erase** (CUInt16Array::iterator it)

- virtual bool **brathl::CInt8Array::Erase** (CInt8Array::iterator it)
- virtual bool **brathl::CUInt8Array::Erase** (CUInt8Array::iterator it)
- virtual bool **brathl::CFloatArray::Erase** (CFloatArray::iterator it)
- virtual bool **brathl::CDoubleArray::Erase** (CDoubleArray::iterator it)
- virtual bool **brathl::CDoublePtrArray::Erase** (CDoublePtrArray::iterator it)
- virtual bool **brathl::CDoublePtrArray::Erase** (int32_t index)
- virtual bool **brathl::CArrayStringMap::Erase** (CArrayStringMap::iterator it)
- virtual bool **brathl::CArrayStringMap::Erase** (const string &key)
- bool **brathl::CObArray::Erase** (CBratObject *ob)
- virtual bool **brathl::CObArray::Erase** (CObArray::iterator it)
- virtual bool **brathl::CObArray::Erase** (int32_t index)
- virtual bool **brathl::CStringMap::Erase** (CStringMap::iterator it)
- virtual bool **brathl::CStringMap::Erase** (const string &key)
- virtual bool **brathl::CIntMap::Erase** (CIntMap::iterator it)
- virtual bool **brathl::CIntMap::Erase** (const string &key)
- virtual bool **brathl::CUIntMap::Erase** (CUIntMap::iterator it)
- virtual bool **brathl::CUIntMap::Erase** (const string &key)
- virtual bool **brathl::CDoubleMap::Erase** (CDoubleMap::iterator it)
- virtual bool **brathl::CDoubleMap::Erase** (const string &key)
- virtual bool **brathl::CObMap::Erase** (CObMap::iterator it)
- virtual bool **brathl::CObMap::Erase** (const string &key)
- virtual bool **brathl::CObIntMap::Erase** (CObIntMap::iterator it)
- virtual bool **brathl::CObIntMap::Erase** (int32_t key)
- virtual bool **brathl::CObDoubleMap::Erase** (CObDoubleMap::iterator it)
- virtual bool **brathl::CObDoubleMap::Erase** (double key)
- virtual bool **brathl::CDoublePtrDoubleMap::Erase** (CDoublePtrDoubleMap::iterator it)
- virtual bool **brathl::CDoublePtrDoubleMap::Erase** (double key)
- virtual bool **brathl::CPtrMap::Erase** (CPtrMap::iterator it)
- virtual bool **brathl::CPtrMap::Erase** (const string &key)
- virtual bool **brathl::CStringList::Exists** (const string &str) const
- virtual bool **brathl::CStringArray::Exists** (const string &str, bool compareNoCase=false) const
- virtual const CStringArray * **brathl::CArrayStringMap::Exists** (const string &key) const
- virtual string **brathl::CStringMap::Exists** (const string &key) const
- virtual int32_t **brathl::CIntMap::Exists** (const string &key) const
- virtual uint32_t **brathl::CUIntMap::Exists** (const string &key) const
- virtual double **brathl::CDoubleMap::Exists** (const string &key) const
- virtual CBratObject * **brathl::CObMap::Exists** (const string &key) const
- virtual CBratObject * **brathl::CObIntMap::Exists** (int32_t key) const
- virtual CBratObject * **brathl::CObDoubleMap::Exists** (double key) const
- virtual DoublePtr * **brathl::CDoublePtrDoubleMap::Exists** (double key) const
- virtual void * **brathl::CPtrMap::Exists** (const string &key) const
- virtual bool **brathl::CStringList::ExistsNoCase** (const string &str) const
- virtual void **brathl::CStringList::ExtractKeys** (const string &str, const string &delim, bool bRemoveAll=true)
- virtual void **brathl::CStringArray::ExtractKeys** (const string &str, const string &delim, bool bRemoveAll=true)
- virtual void **brathl::CStringList::ExtractStrings** (const string &str, const char delim, bool bRemoveAll=true)
- virtual void **brathl::CStringList::ExtractStrings** (const string &str, const string &delim, bool bRemoveAll=true)
- virtual void **brathl::CStringArray::ExtractStrings** (const string &str, const char delim, bool bRemoveAll=true, bool insertUnique=false)
- virtual void **brathl::CStringArray::ExtractStrings** (const string &str, const string &delim, bool bRemoveAll=true, bool insertUnique=false)
- virtual int32_t **brathl::CStringList::FindIndex** (const string &str, bool compareNoCase=false) const
- virtual int32_t **brathl::CStringArray::FindIndex** (const string &str, bool compareNoCase=false) const
- virtual int32_t **brathl::CDoubleArray::FindIndex** (double value) const

- virtual void **bratl::CStringArray::FindIndexes** (const string &str, **CIntArray** &indexes, bool compareNoCase=false) const
- const CArrayDoublePtrArray & **bratl::CMatrixDoublePtr::GetData** ()
- const CArrayDoubleArray & **bratl::CMatrixDouble::GetData** ()
- bool **bratl::CObList::GetDelete** ()
- bool **bratl::CDoublePtrArray::GetDelete** ()
- bool **bratl::CArrayDoublePtrArray::GetDelete** ()
- bool **bratl::CObStack::GetDelete** ()
- bool **bratl::CObArray::GetDelete** ()
- bool **bratl::CObMap::GetDelete** ()
- bool **bratl::CObIntMap::GetDelete** ()
- bool **bratl::CObDoubleMap::GetDelete** ()
- bool **bratl::CDoublePtrDoubleMap::GetDelete** ()
- virtual void **bratl::CStringMap::GetKeys** (CStringArray &keys, bool bRemoveAll=true) const
- virtual void **bratl::CUIntMap::GetKeys** (CStringArray &keys, bool bRemoveAll=true)
- virtual void **bratl::CObMap::GetKeys** (CStringArray &keys, bool bRemoveAll=true, bool bUnique=false)
- virtual void **bratl::CObMap::GetKeys** (CStringList &keys, bool bRemoveAll=true, bool bUnique=false)
- virtual void **bratl::CObIntMap::GetKeys** (CIntArray &keys, bool bRemoveAll=true)
- virtual void **bratl::CObDoubleMap::GetKeys** (CDoubleArray &keys, bool bRemoveAll=true)
- virtual void **bratl::CDoublePtrDoubleMap::GetKeys** (CDoubleArray &keys, bool bRemoveAll=true)
- uint32_t **bratl::CDoublePtrDoubleMap::GetMatrixColDim** (uint32_t row)
- CStringArray * **bratl::CMatrixDoublePtr::GetMatrixDataDimIndexes** ()
- uint32_t **bratl::CDoublePtrArray::GetMatrixDim** (uint32_t row)
- uint32_t **bratl::CArrayDoublePtrArray::GetMatrixDim** (uint32_t row)
- uint32_t **bratl::CMatrixDoublePtr::GetMatrixDimData** (uint32_t row)
- CIntArray * **bratl::CDoublePtrArray::GetMatrixDims** ()
- CIntArray * **bratl::CArrayDoublePtrArray::GetMatrixDims** ()
- CIntArray * **bratl::CDoublePtrDoubleMap::GetMatrixDims** ()
- CIntArray * **bratl::CMatrixDoublePtr::GetMatrixDimsData** ()
- size_t **bratl::CDoublePtrArray::GetMatrixNumberOfDims** ()
- size_t **bratl::CArrayDoublePtrArray::GetMatrixNumberOfDims** ()
- size_t **bratl::CMatrixDoublePtr::GetMatrixNumberOfDimsData** ()
- size_t **bratl::CDoublePtrDoubleMap::GetMatrixNumberOfRows** () const
- virtual uint32_t **bratl::CMatrix::GetMatrixNumberOfValuesData** ()
- uint32_t **bratl::CMatrixDoublePtr::GetMatrixNumberOfValuesData** ()
- uint32_t **bratl::CMatrixDouble::GetMatrixNumberOfValuesData** ()
- void **bratl::CArrayDoublePtrArray::GetMinMaxValues** (double &min, double &max, bool recalc=true)
- void **bratl::CArrayDoubleArray::GetMinMaxValues** (double &min, double &max, bool recalc=true)
- virtual void **bratl::CMatrix::GetMinMaxValues** (double &min, double &max)=0
- virtual void **bratl::CMatrixDoublePtr::GetMinMaxValues** (double &min, double &max)
- virtual void **bratl::CMatrixDouble::GetMinMaxValues** (double &min, double &max)
- string **bratl::CMatrix::GetName** ()
- virtual size_t **bratl::CMatrix::GetNumberOfCols** () const =0
- virtual size_t **bratl::CMatrixDoublePtr::GetNumberOfCols** () const
- virtual size_t **bratl::CMatrixDouble::GetNumberOfCols** () const
- virtual size_t **bratl::CMatrix::GetNumberOfRows** () const =0
- virtual size_t **bratl::CMatrixDoublePtr::GetNumberOfRows** () const
- virtual size_t **bratl::CMatrixDouble::GetNumberOfRows** () const
- virtual uint32_t **bratl::CMatrix::GetNumberOfValues** ()=0
- virtual uint32_t **bratl::CMatrixDoublePtr::GetNumberOfValues** ()
- virtual uint32_t **bratl::CMatrixDouble::GetNumberOfValues** ()
- uint32_t **bratl::CUIntArray::GetProductValues** () const
- uint64_t **bratl::CUInt64Array::GetProductValues** () const
- void **bratl::CFloatArray::GetRange** (float &min, float &max)
- void **bratl::CDoubleArray::GetRange** (double &min, double &max)

- virtual void **brathl::CStringArray::GetValues** (const **CIntArray** &indexes, CStringArray &values) const
- virtual void **brathl::CStringArray::GetValues** (const **CIntArray** &indexes, string &values) const
- string **brathl::CMatrix::GetXName** ()
- string **brathl::CMatrix::GetYName** ()
- virtual void **brathl::CIntArray::IncrementValue** (uint32_t incr=1)
- virtual void **brathl::CInt64Array::IncrementValue** (uint64_t incr=1)
- void **brathl::CArrayDoublePtrArray::Init** ()
- void **brathl::CArrayDoubleArray::Init** ()
- void **brathl::CArrayStringMap::Init** ()
- void **brathl::CArrayDoublePtrArray::InitMatrix** (double initialValue=CTools::m_defaultValueDOUBLE)
- void **brathl::CArrayDoubleArray::InitMatrix** (double initialValue=CTools::m_defaultValueDOUBLE)
- virtual void **brathl::CMatrix::InitMatrix** (double initialValue=CTools::m_defaultValueDOUBLE)=0
- void **brathl::CMatrixDoublePtr::InitMatrix** (double initialValue=CTools::m_defaultValueDOUBLE)
- void **brathl::CMatrixDouble::InitMatrix** (double initialValue=CTools::m_defaultValueDOUBLE)
- void **brathl::CArrayDoublePtrArray::InitMatrixData** (double initialValue=CTools::m_defaultValueDOUBLE)
- void **brathl::CMatrixDoublePtr::InitMatrixDimsData** (const **CUIIntArray** &matrixDims, double initialValue=CTools::m_defaultValueDOUBLE)
- virtual void **brathl::CStringList::Insert** (const **CStringList** &list, bool bEnd=true)
- virtual void **brathl::CStringList::Insert** (const string &str, bool bEnd=true)
- virtual void **brathl::CStringList::Insert** (const CStringArray &vect, bool bEnd=true)
- virtual void **brathl::CStringList::Insert** (const stringarray &vect, bool bEnd=true)
- virtual void **brathl::CStringList::Insert** (const stringlist &lst, bool bEnd=true)
- virtual void **brathl::CIntList::Insert** (const **CIntList** &list, bool bEnd=true)
- virtual void **brathl::CIntList::Insert** (const int value, bool bEnd=true)
- virtual void **brathl::CObList::Insert** (const **CObList** &list, bool bEnd=true)
- virtual void **brathl::CObList::Insert** (CBratObject *ob, bool bEnd=true)
- virtual void **brathl::CStringArray::Insert** (const CStringArray &vect, bool bEnd=true)
- virtual void **brathl::CStringArray::Insert** (const string &str)
- virtual void **brathl::CStringArray::Insert** (const stringarray &vect, bool bEnd=true)
- virtual void **brathl::CStringArray::Insert** (const **CIntArray** &vect)
- virtual void **brathl::CStringArray::Insert** (const **CStringList** &lst)
- virtual void **brathl::CStringArray::Insert** (const stringlist &lst)
- virtual void **brathl::CIntArray::Insert** (const **CIntArray** &vect, bool bEnd=true)
- virtual void **brathl::CIntArray::Insert** (const CStringArray &vect)
- virtual void **brathl::CIntArray::Insert** (int32_t *vect, size_t length)
- virtual void **brathl::CIntArray::Insert** (const int32_t value)
- virtual void **brathl::CInt64Array::Insert** (const **CInt64Array** &vect, bool bEnd=true)
- virtual void **brathl::CInt64Array::Insert** (const CStringArray &vect)
- virtual void **brathl::CInt64Array::Insert** (int64_t *vect, size_t length)
- virtual void **brathl::CInt64Array::Insert** (const int64_t value)
- virtual void **brathl::CUIIntArray::Insert** (**CUIIntArray** *vect, bool bEnd=true)
- virtual void **brathl::CUIIntArray::Insert** (const **CUIIntArray** &vect, bool bEnd=true)
- virtual void **brathl::CUIIntArray::Insert** (const vector< uint32_t > &vect, bool bEnd=true)
- virtual void **brathl::CUIIntArray::Insert** (uint32_t *vect, size_t length)
- virtual void **brathl::CUIIntArray::Insert** (const uint32_t value)
- virtual void **brathl::CUIInt64Array::Insert** (**CUIInt64Array** *vect, bool bEnd=true)
- virtual void **brathl::CUIInt64Array::Insert** (const **CUIInt64Array** &vect, bool bEnd=true)
- virtual void **brathl::CUIInt64Array::Insert** (const vector< uint64_t > &vect, bool bEnd=true)
- virtual void **brathl::CUIInt64Array::Insert** (uint64_t *vect, size_t length)
- virtual void **brathl::CUIInt64Array::Insert** (const uint64_t value)
- virtual void **brathl::CInt16Array::Insert** (const **CInt16Array** &vect, bool bEnd=true)
- virtual void **brathl::CInt16Array::Insert** (const CStringArray &vect)
- virtual void **brathl::CInt16Array::Insert** (int16_t *vect, size_t length)
- virtual void **brathl::CInt16Array::Insert** (const int16_t value)
- virtual void **brathl::CUIInt16Array::Insert** (**CUIInt16Array** *vect, bool bEnd=true)

- virtual void **brathl::CUInt16Array::Insert** (const **CUInt16Array** &vect, bool bEnd=true)
- virtual void **brathl::CUInt16Array::Insert** (const vector< uint16_t > &vect, bool bEnd=true)
- virtual void **brathl::CUInt16Array::Insert** (uint16_t *vect, size_t length)
- virtual void **brathl::CUInt16Array::Insert** (const uint16_t value)
- virtual void **brathl::CInt8Array::Insert** (const **CInt8Array** &vect, bool bEnd=true)
- virtual void **brathl::CInt8Array::Insert** (const CStringArray &vect)
- virtual void **brathl::CInt8Array::Insert** (int8_t *vect, size_t length)
- virtual void **brathl::CInt8Array::Insert** (const int8_t value)
- virtual void **brathl::CUInt8Array::Insert** (**CUInt8Array** *vect, bool bEnd=true)
- virtual void **brathl::CUInt8Array::Insert** (const **CUInt8Array** &vect, bool bEnd=true)
- virtual void **brathl::CUInt8Array::Insert** (const vector< uint8_t > &vect, bool bEnd=true)
- virtual void **brathl::CUInt8Array::Insert** (uint8_t *vect, size_t length)
- virtual void **brathl::CUInt8Array::Insert** (const uint8_t value)
- virtual void **brathl::CFloatArray::Insert** (float *data, int32_t size)
- virtual void **brathl::CFloatArray::Insert** (const **CFloatArray** &vect, bool bEnd=true)
- virtual void **brathl::CFloatArray::Insert** (const **CFloatArray** &vect, int32_t first, int32_t last, bool bEnd=true)
- virtual void **brathl::CFloatArray::Insert** (const float value)
- virtual void **brathl::CFloatArray::Insert** (const int32_t value)
- virtual void **brathl::CFloatArray::Insert** (const uint32_t value)
- virtual void **brathl::CDoubleArray::Insert** (double *data, int32_t size)
- virtual void **brathl::CDoubleArray::Insert** (int32_t *data, int32_t size)
- virtual void **brathl::CDoubleArray::Insert** (uint32_t *data, int32_t size)
- virtual void **brathl::CDoubleArray::Insert** (const **CDoubleArray** &vect, bool bEnd=true)
- virtual void **brathl::CDoubleArray::Insert** (const **CDoubleArray** &vect, int32_t first, int32_t last, bool bEnd=true)
- virtual void **brathl::CDoubleArray::Insert** (const **CUInt8Array** &vect, bool bEnd=true)
- virtual void **brathl::CDoubleArray::Insert** (const **CInt8Array** &vect, bool bEnd=true)
- virtual void **brathl::CDoubleArray::Insert** (const **CInt16Array** &vect, bool bEnd=true)
- virtual void **brathl::CDoubleArray::Insert** (const **CIntArray** &vect, bool bEnd=true)
- virtual void **brathl::CDoubleArray::Insert** (const **CFloatArray** &vect, bool bEnd=true)
- virtual void **brathl::CDoubleArray::Insert** (const CStringArray &vect, bool bEnd=true)
- virtual void **brathl::CDoubleArray::Insert** (const string &vect, const string &delim=",", bool bEnd=true)
- virtual void **brathl::CDoubleArray::Insert** (const double value)
- virtual void **brathl::CDoubleArray::Insert** (const int32_t value)
- virtual void **brathl::CDoubleArray::Insert** (const uint32_t value)
- virtual void **brathl::CDoubleArray::Insert** (const int16_t value)
- virtual void **brathl::CDoubleArray::Insert** (const uint16_t value)
- virtual void **brathl::CDoubleArray::Insert** (const int8_t value)
- virtual void **brathl::CDoubleArray::Insert** (const uint8_t value)
- virtual void **brathl::CDoublePtrArray::Insert** (DoublePtr ob)
- virtual CStringArray * **brathl::CArrayStringMap::Insert** (const string &key, const CStringArray &str, bool withExcept=true)
- virtual void **brathl::CObArray::Insert** (const **CObArray** &vect)
- virtual void **brathl::CObArray::Insert** (CBratObject *ob)
- virtual string **brathl::CStringMap::Insert** (const string &key, const string &str, bool withExcept=true)
- virtual void **brathl::CStringMap::Insert** (const **CStringMap** &strmap, bool withExcept=true)
- virtual int32_t **brathl::CIntMap::Insert** (const string &key, int32_t value, bool withExcept=true)
- virtual void **brathl::CIntMap::Insert** (const **CIntMap** &m, bool bRemoveAll=true, bool withExcept=true)
- virtual void **brathl::CIntMap::Insert** (const CStringArray &keys, const **CIntArray** &values, bool bRemoveAll=true, bool withExcept=true)
- virtual uint32_t **brathl::CUIntMap::Insert** (const string &key, uint32_t value, bool withExcept=true)
- virtual void **brathl::CUIntMap::Insert** (const **CUIntMap** &m, bool bRemoveAll=true, bool withExcept=true)
- virtual void **brathl::CUIntMap::Insert** (const CStringArray &keys, uint32_t initValue, bool bRemoveAll=true, bool withExcept=true)

- virtual void **brathl::CUIntMap::Insert** (const CStringArray &keys, const **CUIntArray** &values, bool bRemoveAll=true, bool withExcept=true)
- virtual void **brathl::CUIntMap::Insert** (const CStringArray &keys, bool bRemoveAll=true, bool withExcept=true)
- virtual double **brathl::CDoubleMap::Insert** (const string &key, double value, bool withExcept=true)
- virtual CBratObject * **brathl::CObMap::Insert** (const string &key, CBratObject *ob, bool withExcept=true)
- virtual void **brathl::CObMap::Insert** (const **CObMap** &obMap, bool withExcept=true)
- virtual CBratObject * **brathl::CObIntMap::Insert** (int32_t key, CBratObject *ob, bool withExcept=true)
- virtual void **brathl::CObIntMap::Insert** (const **CObIntMap** &obMap, bool withExcept=true)
- virtual CBratObject * **brathl::CObDoubleMap::Insert** (double key, CBratObject *ob, bool withExcept=true)
- virtual void **brathl::CObDoubleMap::Insert** (const **CObDoubleMap** &obMap, bool withExcept=true)
- virtual DoublePtr * **brathl::CDoublePtrDoubleMap::Insert** (double key, DoublePtr *ob, bool withExcept=true)
- virtual DoublePtr * **brathl::CDoublePtrDoubleMap::Insert** (double key, double initialValue=CTools::m_defaultValueDOUBLE)
- virtual void * **brathl::CPtrMap::Insert** (const string &key, void *ptr, bool withExcept=true)
- virtual void **brathl::CPtrMap::Insert** (const **CPtrMap** &ptrMap, bool withExcept=true)
- virtual CStringArray::iterator **brathl::CStringArray::InsertAt** (CStringArray::iterator where, const string &str)
- virtual CStringArray::iterator **brathl::CStringArray::InsertAt** (int32_t index, const string &str)
- virtual CIntArray::iterator **brathl::CIntArray::InsertAt** (CIntArray::iterator where, const int32_t value)
- virtual CIntArray::iterator **brathl::CIntArray::InsertAt** (int32_t index, const int32_t value)
- virtual CInt64Array::iterator **brathl::CInt64Array::InsertAt** (CInt64Array::iterator where, const int64_t value)
- virtual CInt64Array::iterator **brathl::CInt64Array::InsertAt** (size_t index, const int64_t value)
- virtual CUIntArray::iterator **brathl::CUIntArray::InsertAt** (CUIntArray::iterator where, const uint32_t value)
- virtual CUIntArray::iterator **brathl::CUIntArray::InsertAt** (int32_t index, const uint32_t value)
- virtual CUInt64Array::iterator **brathl::CUInt64Array::InsertAt** (CUInt64Array::iterator where, const uint64_t value)
- virtual CUInt64Array::iterator **brathl::CUInt64Array::InsertAt** (size_t index, const uint64_t value)
- virtual CInt16Array::iterator **brathl::CInt16Array::InsertAt** (CInt16Array::iterator where, const int16_t value)
- virtual CInt16Array::iterator **brathl::CInt16Array::InsertAt** (int32_t index, const int16_t value)
- virtual CUInt16Array::iterator **brathl::CUInt16Array::InsertAt** (CUInt16Array::iterator where, const uint16_t value)
- virtual CUInt16Array::iterator **brathl::CUInt16Array::InsertAt** (int32_t index, const uint16_t value)
- virtual CInt8Array::iterator **brathl::CInt8Array::InsertAt** (CInt8Array::iterator where, const int8_t value)
- virtual CInt8Array::iterator **brathl::CInt8Array::InsertAt** (int32_t index, const int8_t value)
- virtual CUInt8Array::iterator **brathl::CUInt8Array::InsertAt** (CUInt8Array::iterator where, const uint8_t value)
- virtual CUInt8Array::iterator **brathl::CUInt8Array::InsertAt** (int32_t index, const uint8_t value)
- virtual CFloatArray::iterator **brathl::CFloatArray::InsertAt** (CFloatArray::iterator where, const float value)
- virtual CFloatArray::iterator **brathl::CFloatArray::InsertAt** (int32_t index, const float value)
- virtual CDoubleArray::iterator **brathl::CDoubleArray::InsertAt** (CDoubleArray::iterator where, const double value)
- virtual CDoubleArray::iterator **brathl::CDoubleArray::InsertAt** (int32_t index, const double value)
- virtual CDoublePtrArray::iterator **brathl::CDoublePtrArray::InsertAt** (CDoublePtrArray::iterator where, DoublePtr ob)
- virtual CObArray::iterator **brathl::CObArray::InsertAt** (CObArray::iterator where, CBratObject *ob)
- virtual void **brathl::CStringList::InsertUnique** (const string &str, bool bEnd=true)
- virtual void **brathl::CStringList::InsertUnique** (const **CStringList** &lst, bool bEnd=true)
- virtual void **brathl::CStringList::InsertUnique** (const CStringArray *vect, bool bEnd=true)
- virtual void **brathl::CStringList::InsertUnique** (const CStringArray &vect, bool bEnd=true)
- virtual void **brathl::CStringList::InsertUnique** (const stringarray &vect, bool bEnd=true)
- virtual void **brathl::CStringList::InsertUnique** (const stringlist &lst, bool bEnd=true)
- virtual void **brathl::CStringArray::InsertUnique** (const string &str)
- virtual void **brathl::CStringArray::InsertUnique** (const CStringArray *vect)
- virtual void **brathl::CStringArray::InsertUnique** (const CStringArray &vect)
- virtual void **brathl::CStringArray::InsertUnique** (const **CStringList** &lst)

- virtual void **brathl::CStringArray::InsertUnique** (const stringarray &vect)
- virtual void **brathl::CStringArray::InsertUnique** (const stringlist &lst)
- virtual bool **brathl::CStringList::Intersect** (const **CStringList** &array, **CStringList** &intersect) const
- virtual bool **brathl::CStringArray::Intersect** (const string &str, CStringArray &intersect, bool compareNoCase=false) const
- virtual bool **brathl::CStringArray::Intersect** (const CStringArray &array, CStringArray &intersect, bool compareNoCase=false) const
- virtual bool **brathl::CStringArray::Intersect** (const string &str, **CUIntArray** &intersect, bool compareNoCase=false) const
- virtual bool **brathl::CStringArray::Intersect** (const CStringArray &array, **CUIntArray** &intersect, bool compareNoCase=false) const
- virtual bool **brathl::CIntArray::Intersect** (const **CIntArray** &array, **CIntArray** &intersect) const
- virtual bool **brathl::CInt64Array::Intersect** (const **CInt64Array** &array, **CInt64Array** &intersect) const
- virtual bool **brathl::CUIntArray::Intersect** (const **CUIntArray** &array, **CUIntArray** &intersect) const
- virtual bool **brathl::CUInt64Array::Intersect** (const **CUInt64Array** &array, **CUInt64Array** &intersect) const
- virtual bool **brathl::CInt16Array::Intersect** (const **CInt16Array** &array, **CInt16Array** &intersect) const
- virtual bool **brathl::CUInt16Array::Intersect** (const **CUInt16Array** &array, **CUInt16Array** &intersect) const
- virtual bool **brathl::CInt8Array::Intersect** (const **CInt8Array** &array, **CInt8Array** &intersect) const
- virtual bool **brathl::CUInt8Array::Intersect** (const **CUInt8Array** &array, **CUInt8Array** &intersect) const
- virtual bool **brathl::CFloatArray::Intersect** (const **CFloatArray** &array, **CFloatArray** &intersect) const
- virtual bool **brathl::CDoubleArray::Intersect** (const **CDoubleArray** &array, **CDoubleArray** &intersect) const

- virtual bool **brathl::CMatrix::IsMatrixDataSet** ()
- bool **brathl::CMatrixDoublePtr::IsMatrixDataSet** ()
- virtual string **brathl::CStringMap::IsValue** (const string &value)
- DoublePtr **brathl::CDoublePtrArray::NewMatrix** (double initialValue=CTools::m_defaultValueDOUBLE)
- DoublePtr **brathl::CArrayDoublePtrArray::NewMatrix** (double initialValue=CTools::m_defaultValueDOUBLE)
- DoublePtr * **brathl::CDoublePtrDoubleMap::NewMatrix** (double initialValue=CTools::m_defaultValueDOUBLE)
- DoublePtr **brathl::CMatrixDoublePtr::NewMatrixData** (double initialValue=CTools::m_defaultValueDOUBLE)
- virtual bool **brathl::CStringArray::operator!=** (const CStringArray &vect)
- virtual bool **brathl::CUIntArray::operator!=** (const **CUIntArray** &vect)
- virtual bool **brathl::CUInt64Array::operator!=** (const **CUInt64Array** &vect)
- virtual bool **brathl::CUInt16Array::operator!=** (const **CUInt16Array** &vect)
- virtual bool **brathl::CUInt8Array::operator!=** (const **CUInt8Array** &vect)
- virtual bool **brathl::CDoubleArray::operator!=** (const **CDoubleArray** &vect)
- virtual DoublePtr **brathl::CMatrix::operator()** (uint32_t i, uint32_t j)=0
- virtual DoublePtr **brathl::CMatrix::operator()** (uint32_t i, uint32_t j) const =0
- virtual DoublePtr **brathl::CMatrixDoublePtr::operator()** (uint32_t i, uint32_t j)
- virtual DoublePtr **brathl::CMatrixDoublePtr::operator()** (uint32_t i, uint32_t j) const
- virtual DoublePtr **brathl::CMatrixDouble::operator()** (uint32_t i, uint32_t j)
- virtual DoublePtr **brathl::CMatrixDouble::operator()** (uint32_t i, uint32_t j) const
- virtual const **CStringList** & **brathl::CStringList::operator=** (const **CStringList** &lst)
- virtual const **CStringList** & **brathl::CStringList::operator=** (const CStringArray &vect)
- virtual const **CStringList** & **brathl::CStringList::operator=** (const stringarray &vect)
- virtual const **CStringList** & **brathl::CStringList::operator=** (const stringlist &lst)
- const **CIntList** & **brathl::CIntList::operator=** (const **CIntList** &lst)
- virtual const **CObList** & **brathl::CObList::operator=** (const **CObList** &lst)
- virtual const CStringArray & **brathl::CStringArray::operator=** (const CStringArray &vect)
- virtual const CStringArray & **brathl::CStringArray::operator=** (const **CStringList** &lst)
- virtual const CStringArray & **brathl::CStringArray::operator=** (const stringarray &vect)
- virtual const CStringArray & **brathl::CStringArray::operator=** (const stringlist &lst)
- virtual const **CIntArray** & **brathl::CIntArray::operator=** (const **CIntArray** &vect)

- virtual const **CInt64Array** & **brathl::CInt64Array::operator=** (const **CInt64Array** &vect)
- virtual const **CUIntArray** & **brathl::CUIntArray::operator=** (const **CUIntArray** &vect)
- virtual const **CUInt64Array** & **brathl::CUInt64Array::operator=** (const **CUInt64Array** &vect)
- virtual const **CInt16Array** & **brathl::CInt16Array::operator=** (const **CInt16Array** &vect)
- virtual const **CUInt16Array** & **brathl::CUInt16Array::operator=** (const **CUInt16Array** &vect)
- virtual const **CInt8Array** & **brathl::CInt8Array::operator=** (const **CInt8Array** &vect)
- virtual const **CUInt8Array** & **brathl::CUInt8Array::operator=** (const **CUInt8Array** &vect)
- virtual const **CFloatArray** & **brathl::CFloatArray::operator=** (const **CFloatArray** &vect)
- virtual const **CDoubleArray** & **brathl::CDoubleArray::operator=** (const **CDoubleArray** &vect)
- virtual const **CArrayDoublePtrArray** & **brathl::CArrayDoublePtrArray::operator=** (const **CArrayDoublePtrArray** &m)
- virtual const **CArrayDoubleArray** & **brathl::CArrayDoubleArray::operator=** (const **CArrayDoubleArray** &m)
- virtual const **CArrayStringMap** & **brathl::CArrayStringMap::operator=** (const **CArrayStringMap** &a)
- virtual const **CDoubleArrayOb** & **brathl::CDoubleArrayOb::operator=** (const **CDoubleArrayOb** &vect)
- virtual const **CObArray** & **brathl::CObArray::operator=** (const **CObArray** &lst)
- virtual const **CObArrayOb** & **brathl::CObArrayOb::operator=** (const **CObArrayOb** &vect)
- virtual const **CObMap** & **brathl::CObMap::operator=** (const **CObMap** &obMap)
- virtual const **CObIntMap** & **brathl::CObIntMap::operator=** (const **CObIntMap** &obMap)
- virtual const **CObDoubleMap** & **brathl::CObDoubleMap::operator=** (const **CObDoubleMap** &obMap)
- const **CMatrix** & **brathl::CMatrix::operator=** (const **CMatrix** &m)
- const **CMatrixDoublePtr** & **brathl::CMatrixDoublePtr::operator=** (const **CMatrixDoublePtr** &m)
- const **CMatrixDouble** & **brathl::CMatrixDouble::operator=** (const **CMatrixDouble** &m)
- virtual bool **brathl::CStringArray::operator==** (const **CStringArray** &vect)
- virtual bool **brathl::CIntArray::operator==** (const **CIntArray** &vect)
- virtual bool **brathl::CInt64Array::operator==** (const **CInt64Array** &vect)
- virtual bool **brathl::CUIntArray::operator==** (const **CUIntArray** &vect)
- virtual bool **brathl::CUInt64Array::operator==** (const **CUInt64Array** &vect)
- virtual bool **brathl::CUInt16Array::operator==** (const **CUInt16Array** &vect)
- virtual bool **brathl::CUInt8Array::operator==** (const **CUInt8Array** &vect)
- virtual bool **brathl::CDoubleArray::operator==** (const **CDoubleArray** &vect)
- virtual int32_t **brathl::CIntMap::operator[]** (const string &key)
- virtual uint32_t **brathl::CUIntMap::operator[]** (const string &key)
- virtual double **brathl::CDoubleMap::operator[]** (const string &key)
- virtual CBratObject * **brathl::CObMap::operator[]** (const string &key)
- virtual CBratObject * **brathl::CObIntMap::operator[]** (int32_t key)
- virtual CBratObject * **brathl::CObDoubleMap::operator[]** (double key)
- virtual DoublePtr * **brathl::CDoublePtrDoubleMap::operator[]** (double key)
- virtual void * **brathl::CPtrMap::operator[]** (const string &key)
- virtual doubleptrarray & **brathl::CMatrixDoublePtr::operator[]** (const uint32_t &i)
- virtual const doubleptrarray & **brathl::CMatrixDoublePtr::operator[]** (const uint32_t &i) const
- virtual doublearray & **brathl::CMatrixDouble::operator[]** (const uint32_t &i)
- virtual const doublearray & **brathl::CMatrixDouble::operator[]** (const uint32_t &i) const
- virtual void **brathl::CObStack::Pop** ()
- virtual bool **brathl::CObList::PopBack** ()
- virtual bool **brathl::CDoublePtrArray::PopBack** ()
- virtual bool **brathl::CObArray::PopBack** ()
- virtual void **brathl::CObStack::Push** (CBratObject *ob)
- virtual bool **brathl::CStringArray::Remove** (const string &array, bool compareNoCase=false)
- virtual bool **brathl::CStringArray::Remove** (const **CStringArray** &array, bool compareNoCase=false)
- virtual void **brathl::CArrayDoublePtrArray::Remove** (doubleptrarray &vect)
- virtual void **brathl::CStringList::RemoveAll** ()
- virtual void **brathl::CIntList::RemoveAll** ()
- virtual void **brathl::CObList::RemoveAll** ()
- virtual void **brathl::CStringArray::RemoveAll** ()
- virtual void **brathl::CIntArray::RemoveAll** ()

- virtual void **brathl::CInt64Array::RemoveAll** ()
- virtual void **brathl::CUIntArray::RemoveAll** ()
- virtual void **brathl::CUInt64Array::RemoveAll** ()
- virtual void **brathl::CInt16Array::RemoveAll** ()
- virtual void **brathl::CUInt16Array::RemoveAll** ()
- virtual void **brathl::CInt8Array::RemoveAll** ()
- virtual void **brathl::CUInt8Array::RemoveAll** ()
- virtual void **brathl::CFloatArray::RemoveAll** ()
- virtual void **brathl::CDoubleArray::RemoveAll** ()
- virtual void **brathl::CDoublePtrArray::RemoveAll** ()
- virtual void **brathl::CArrayDoublePtrArray::RemoveAll** ()
- virtual void **brathl::CArrayDoubleArray::RemoveAll** ()
- virtual void **brathl::CArrayStringMap::RemoveAll** ()
- virtual void **brathl::CObStack::RemoveAll** ()
- virtual void **brathl::CObArray::RemoveAll** ()
- virtual void **brathl::CStringMap::RemoveAll** ()
- virtual void **brathl::CIntMap::RemoveAll** ()
- virtual void **brathl::CUIntMap::RemoveAll** ()
- virtual void **brathl::CDoubleMap::RemoveAll** ()
- virtual void **brathl::CObMap::RemoveAll** ()
- virtual void **brathl::CObIntMap::RemoveAll** ()
- virtual void **brathl::CObDoubleMap::RemoveAll** ()
- virtual void **brathl::CDoublePtrDoubleMap::RemoveAll** ()
- virtual void **brathl::CPtrMap::RemoveAll** ()
- bool **brathl::CObMap::RenameKey** (const string &oldKey, const string &newKey)
- bool **brathl::CObIntMap::RenameKey** (int32_t oldKey, int32_t newKey)
- bool **brathl::CObDoubleMap::RenameKey** (double oldKey, double newKey)
- bool **brathl::CDoublePtrDoubleMap::RenameKey** (double oldKey, double newKey)
- virtual void **brathl::CStringArray::Replace** (const CStringArray &findString, const string &replaceBy, CStringArray &replaced, bool compareNoCase=false, bool insertUnique=false) const
- virtual void **brathl::CStringArray::Replace** (const string &findString, const string &replaceBy, CStringArray &replaced, bool compareNoCase=false, bool insertUnique=false) const
- virtual CStringArray::iterator **brathl::CStringArray::ReplaceAt** (int32_t index, const string &str)
- virtual CStringArray::iterator **brathl::CStringArray::ReplaceAt** (uint32_t index, const string &str)
- virtual CStringArray::iterator **brathl::CStringArray::ReplaceAt** (CStringArray::iterator where, const string &str)
- virtual CIntArray::iterator **brathl::CIntArray::ReplaceAt** (CIntArray::iterator where, const int32_t value)
- virtual CIntArray::iterator **brathl::CIntArray::ReplaceAt** (int32_t index, const int32_t value)
- virtual CInt64Array::iterator **brathl::CInt64Array::ReplaceAt** (CInt64Array::iterator where, const int64_t value)
- virtual CInt64Array::iterator **brathl::CInt64Array::ReplaceAt** (size_t index, const int64_t value)
- virtual CUIntArray::iterator **brathl::CUIntArray::ReplaceAt** (CUIntArray::iterator where, const uint32_t value)
- virtual CUIntArray::iterator **brathl::CUIntArray::ReplaceAt** (int32_t index, const uint32_t value)
- virtual CUInt64Array::iterator **brathl::CUInt64Array::ReplaceAt** (CUInt64Array::iterator where, const uint64_t value)
- virtual CUInt64Array::iterator **brathl::CUInt64Array::ReplaceAt** (size_t index, const uint64_t value)
- virtual CInt16Array::iterator **brathl::CInt16Array::ReplaceAt** (CInt16Array::iterator where, const int16_t value)
- virtual CInt16Array::iterator **brathl::CInt16Array::ReplaceAt** (int32_t index, const int16_t value)
- virtual CUInt16Array::iterator **brathl::CUInt16Array::ReplaceAt** (CUInt16Array::iterator where, const uint16_t value)
- virtual CUInt16Array::iterator **brathl::CUInt16Array::ReplaceAt** (int32_t index, const uint16_t value)
- virtual CInt8Array::iterator **brathl::CInt8Array::ReplaceAt** (CInt8Array::iterator where, const int8_t value)
- virtual CInt8Array::iterator **brathl::CInt8Array::ReplaceAt** (int32_t index, const int8_t value)

- virtual CUInt8Array::iterator **brathl::CUInt8Array::ReplaceAt** (CUInt8Array::iterator where, const uint8_t value)
- virtual CUInt8Array::iterator **brathl::CUInt8Array::ReplaceAt** (int32_t index, const uint8_t value)
- virtual CFloatArray::iterator **brathl::CFloatArray::ReplaceAt** (CFloatArray::iterator where, const float value)
- virtual CFloatArray::iterator **brathl::CFloatArray::ReplaceAt** (int32_t index, const float value)
- virtual CDoubleArray::iterator **brathl::CDoubleArray::ReplaceAt** (CDoubleArray::iterator where, const double value)
- virtual CDoubleArray::iterator **brathl::CDoubleArray::ReplaceAt** (int32_t index, const double value)
- virtual CDoublePtrArray::iterator **brathl::CDoublePtrArray::ReplaceAt** (CDoublePtrArray::iterator where, DoublePtr ob)
- virtual CObArray::iterator **brathl::CObArray::ReplaceAt** (CObArray::iterator where, CBratObject *ob)
- void **brathl::CArrayDoublePtrArray::ResizeRC** (uint32_t nrows, uint32_t ncols)
- void **brathl::CArrayDoubleArray::ResizeRC** (uint32_t nrows, uint32_t ncols)
- virtual void **brathl::CMatrix::ScaleDownData** (double scaleFactor, double addOffset, double defaultValue=CTools::m_defaultValueDOUBLE)=0
- virtual void **brathl::CMatrixDoublePtr::ScaleDownData** (double scaleFactor, double addOffset, double defaultValue=CTools::m_defaultValueDOUBLE)
- virtual void **brathl::CMatrixDouble::ScaleDownData** (double scaleFactor, double addOffset, double defaultValue=CTools::m_defaultValueDOUBLE)
- virtual void **brathl::CMatrix::ScaleUpData** (double scaleFactor, double addOffset, double defaultValue=CTools::m_defaultValueDOUBLE)=0
- virtual void **brathl::CMatrixDoublePtr::ScaleUpData** (double scaleFactor, double addOffset, double defaultValue=CTools::m_defaultValueDOUBLE)
- virtual void **brathl::CMatrixDouble::ScaleUpData** (double scaleFactor, double addOffset, double defaultValue=CTools::m_defaultValueDOUBLE)
- void **brathl::CArrayDoublePtrArray::Set** (const CArrayDoublePtrArray &m)
- void **brathl::CArrayDoubleArray::Set** (const CArrayDoubleArray &m)
- virtual void **brathl::CArrayStringMap::Set** (const CArrayStringMap &a)
- virtual void **brathl::CMatrix::Set** (const CMatrix &m)
- virtual void **brathl::CMatrix::Set** (uint32_t &row, uint32_t &col, DoublePtr x)=0
- void **brathl::CMatrixDoublePtr::Set** (uint32_t &row, uint32_t &col, DoublePtr x)
- void **brathl::CMatrixDoublePtr::Set** (const CMatrixDoublePtr &m)
- void **brathl::CMatrixDouble::Set** (uint32_t &row, uint32_t &col, DoublePtr x)
- void **brathl::CMatrixDouble::Set** (const CMatrixDouble &m)
- void **brathl::CObList::SetDelete** (bool value)
- void **brathl::CDoublePtrArray::SetDelete** (bool value)
- void **brathl::CArrayDoublePtrArray::SetDelete** (bool value)
- void **brathl::CObStack::SetDelete** (bool value)
- void **brathl::CObArray::SetDelete** (bool value)
- void **brathl::CObMap::SetDelete** (bool value)
- void **brathl::CObIntMap::SetDelete** (bool value)
- void **brathl::CObDoubleMap::SetDelete** (bool value)
- void **brathl::CDoublePtrDoubleMap::SetDelete** (bool value)
- void **brathl::CMatrixDoublePtr::SetMatrixDataDimIndexes** (const CStringArray &m)
- void **brathl::CDoublePtrArray::SetMatrixDims** (const CUIntArray &matrixDims)
- void **brathl::CArrayDoublePtrArray::SetMatrixDims** (const CUIntArray &matrixDims)
- void **brathl::CDoublePtrDoubleMap::SetMatrixDims** (const CUIntArray &matrixDims)
- void **brathl::CMatrixDoublePtr::SetMatrixDimsData** (const CUIntArray &matrixDims)
- void **brathl::CMatrixDoublePtr::SetMatrixDimsData** (uint32_t nbValues)
- void **brathl::CMatrix::SetName** (const string &value)
- void **brathl::CMatrix::SetXName** (const string &value)
- void **brathl::CMatrix::SetYName** (const string &value)
- virtual int32_t * **brathl::CIntArray::ToArray** ()
- virtual int64_t * **brathl::CInt64Array::ToArray** ()
- virtual uint32_t * **brathl::CUIntArray::ToArray** ()

- virtual uint64_t * **brathl::CUInt64Array::ToArray** ()
- virtual int16_t * **brathl::CInt16Array::ToArray** ()
- virtual uint16_t * **brathl::CUInt16Array::ToArray** ()
- virtual int8_t * **brathl::CInt8Array::ToArray** ()
- virtual uint8_t * **brathl::CUInt8Array::ToArray** ()
- float * **brathl::CFloatArray::ToArray** ()
- double * **brathl::CDoubleArray::ToArray** ()
- virtual void **brathl::CObMap::ToArray** (CObArray &obArray)
- virtual int32_t * **brathl::CUIntArray::ToIntArray** ()
- virtual int16_t * **brathl::CUInt16Array::ToIntArray** ()
- virtual int8_t * **brathl::CUInt8Array::ToIntArray** ()
- virtual CBratObject * **brathl::CObStack::Top** ()
- virtual size_t * **brathl::CUIntArray::ToSizeTArray** ()
- virtual string **brathl::CStringList::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CStringArray::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CIntArray::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CInt64Array::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CUIntArray::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CUInt64Array::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CInt16Array::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CUInt16Array::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CInt8Array::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CUInt8Array::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CFloatArray::ToString** (const string &delim=",", bool useBracket=true) const
- virtual string **brathl::CDoubleArray::ToString** (const string &delim=",", bool useBracket=true) const
- virtual **brathl::CArrayDoubleArray::~CArrayDoubleArray** ()

Destructor.

- virtual **brathl::CArrayDoublePtrArray::~CArrayDoublePtrArray** ()

Destructor.

- virtual **brathl::CArrayStringMap::~CArrayStringMap** ()

CStringMap (p. 320) dtor.

- virtual **brathl::CDoubleArray::~CDoubleArray** ()

Destructor.

- virtual **brathl::CDoubleMap::~CDoubleMap** ()

CDoubleMap (p. 196) dtor.

- virtual **brathl::CDoublePtrArray::~CDoublePtrArray** ()

Destructor.

- virtual **brathl::CDoublePtrDoubleMap::~CDoublePtrDoubleMap** ()

CDoublePtrDoubleMap (p. 198) dtor.

- virtual **brathl::CFloatArray::~CFloatArray** ()

Destructor.

- virtual **brathl::CInt16Array::~CInt16Array** ()

Destructor.

- virtual **brathl::CInt64Array::~CInt64Array** ()

Destructor.

- virtual **brathl::CInt8Array::~CInt8Array** ()

Destructor.

- virtual **brathl::CIntArray::~CIntArray** ()

Destructor.

- virtual **brathl::CIntList::~CIntList** ()

Destructor.

- virtual **brathl::CIntMap::~CIntMap** ()

- *CIntMap* (p. 257) dtor.
- virtual **bratl::CObArray::~~CObArray** ()
Destructor.
- virtual **bratl::CObDoubleMap::~~CObDoubleMap** ()
CObMap (p. 274) dtor.
- virtual **bratl::COblntMap::~~COblntMap** ()
CObMap (p. 274) dtor.
- virtual **bratl::CObList::~~CObList** ()
Destructor.
- virtual **bratl::CObMap::~~CObMap** ()
CObMap (p. 274) dtor.
- virtual **bratl::CObStack::~~CObStack** ()
Destructor.
- virtual **bratl::CPtrMap::~~CPtrMap** ()
CPtrMap (p. 315) dtor.
- virtual **bratl::CStringArray::~~CStringArray** ()
Destructor.
- virtual **bratl::CStringList::~~CStringList** ()
Destructor.
- virtual **bratl::CStringMap::~~CStringMap** ()
CStringMap (p. 320) dtor.
- virtual **bratl::CUInt16Array::~~CUInt16Array** ()
Destructor.
- virtual **bratl::CUInt64Array::~~CUInt64Array** ()
Destructor.
- virtual **bratl::CUInt8Array::~~CUInt8Array** ()
Destructor.
- virtual **bratl::CUIntArray::~~CUIntArray** ()
Destructor.
- virtual **bratl::CUIntMap::~~CUIntMap** ()
CUIntMap (p. 352) dtor.

Variables

- const string **bratl::GENERIC_NETCDF_TYPE** = "Generic NetCdf"
- bool **bratl::CObList::m_bDelete**
- bool **bratl::CDoublePtrArray::m_bDelete**
- bool **bratl::CArrayDoublePtrArray::m_bDelete**
- bool **bratl::CObStack::m_bDelete**
Dump fonction.
- bool **bratl::CObArray::m_bDelete**
- bool **bratl::CObMap::m_bDelete**
- bool **bratl::COblntMap::m_bDelete**
- bool **bratl::CObDoubleMap::m_bDelete**
- bool **bratl::CDoublePtrDoubleMap::m_bDelete**
- bool **bratl::CPtrMap::m_bDelete**
- CArrayDoublePtrArray **bratl::CMatrixDoublePtr::m_data**
- CStringArray **bratl::CMatrixDoublePtr::m_matrixDataDimIndexes**
- CUIntArray **bratl::CDoublePtrArray::m_matrixDims**
- CUIntArray **bratl::CArrayDoublePtrArray::m_matrixDims**
- CUIntArray **bratl::CDoublePtrDoubleMap::m_matrixDims**
- double **bratl::CArrayDoublePtrArray::m_maxValue**

- double **brathl::CArrayDoubleArray::m_maxValue**
- double **brathl::CArrayDoublePtrArray::m_minValue**
- double **brathl::CArrayDoubleArray::m_minValue**
- const string **brathl::NETCDF_CF_PRODUCT_CLASS** = "NETCDF_CF"
- const string **brathl::NETCDF_PRODUCT_CLASS** = "NETCDF"
- const string **brathl::UNKNOWN_PRODUCT_CLASS** = "UNKNOWN"
- const string **brathl::YFX_NETCDF_TYPE** = "Y=F(X)"
- const string **brathl::ZFX_Y_NETCDF_TYPE** = "Z=F(X,Y)"

5.5.1 Detailed Description

5.5.2 Macro Definition Documentation

5.5.2.1 #define FILL_VALUE_ATTR "_FillValue"

NetCDF files access.

Version

1.0

5.5.3 Typedef Documentation

5.5.3.1 typedef vector<doublearray> brathl::arraydoublearray

An array (vector) of vector of double

Version

1.0

Creates a type name for array of double array

5.5.3.2 typedef vector<doubleptrarray> brathl::arraydoubleptrarray

An array (vector) of vector of double pointer

Version

1.0

Creates a type name for array of DoublePtr array

5.5.3.3 typedef map<string, CStringArray> brathl::maparraystring

a set of array string value management classes.

Version

1.0

Creates a type name for map of string array

5.5.4 Function Documentation

5.5.4.1 CExternalFiles * brathl::BuildExistingExternalFileKind (const string & Name)

External files access.

Version

1.0

5.5.4.2 CInternalFiles * brathl::BuildExistingInternalFileKind (const string & name, const CStringArray * fieldNames = NULL)

Internal files access.

Version

1.0

References BRATHL_ERROR, and brathl::CTools::Format().

5.5.4.3 brathl::CDoubleArray::CDoubleArray (const CDoubleArray & vect)

Creates new **CDoubleArray** (p. 195) object from another **CDoubleArray** (p. 195)

Parameters

<i>vect</i>	[in] : array to be copied
-------------	---------------------------

5.5.4.4 brathl::CFloatArray::CFloatArray (const CFloatArray & vect)

Creates new **CFloatArray** (p. 248) object from another **CFloatArray** (p. 248)

Parameters

<i>vect</i>	[in] : array to be copied
-------------	---------------------------

5.5.4.5 brathl::CInt16Array::CInt16Array (const CInt16Array & vect)

Creates new **CInt16Array** (p. 249) object from another **CStringList** (p. 319)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.6 brathl::CInt64Array::CInt64Array (const CInt64Array & v)

Creates new **CInt64Array** (p. 250) object from another **CInt64Array** (p. 250)

Parameters

<i>v</i>	[in] : vector to be copied
----------	----------------------------

5.5.4.7 brathl::CInt8Array::CInt8Array (const CInt8Array & vect)

Creates new **CInt8Array** (p. 251) object from another **CStringList** (p. 319)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.8 brathl::CIntArray::CIntArray (const CIntArray & vect)

Creates new **CIntArray** (p. 251) object from another **CStringList** (p. 319)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.9 brathl::CIntList::CIntList (const CIntList & *list*)

Creates new **CIntList** (p. 257) object from another **CStringList** (p. 319)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.10 brathl::CObArray::CObArray (const CObArray & *vect*)

Creates new **CObArray** (p. 270) object from another **CObArray** (p. 270)

Parameters

<i>vect</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.11 brathl::CObList::CObList (const CObList & *lst*)

Creates new **CObList** (p. 272) object from another **CStringList** (p. 319)

Parameters

<i>lst</i>	[in] : list to be copied
------------	--------------------------

5.5.4.12 brathl::CStringArray::CStringArray (const CStringArray & *vect*)

Creates new **CStringArray** object from another **CStringList** (p. 319)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.13 brathl::CStringList::CStringList (const CStringList & *list*)

Creates new **CStringList** (p. 319) object from another **CStringList** (p. 319)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.14 brathl::CUInt16Array::CUInt16Array (const CUInt16Array & *vect*)

Creates new **CUInt16Array** (p. 348) object from another **CStringList** (p. 319)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.15 brathl::CUInt64Array::CUInt64Array (const CUInt64Array & *vect*)

Creates new **CUInt64Array** (p. 349) object from another **CStringList** (p. 319)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.16 `brat::CUInt8Array::CUInt8Array (const CUInt8Array & vect)`

Creates new **CUInt8Array** (p. 350) object from another **CStringList** (p. 319)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.17 `brat::CUIntArray::CUIntArray (const CUIntArray & vect)`

Creates new **CUIntArray** (p. 351) object from another **CStringList** (p. 319)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

5.5.4.18 `bool brat::CObList::Erase (CBratObject * ob)`

Delete an element referenced by ob

Returns

true if no error, otherwise false

5.5.4.19 `bool brat::CObList::Erase (CObList::iterator it) [virtual]`

Delete an element referenced by it

Returns

true if no error, otherwise false

5.5.4.20 `bool brat::CDoublePtrArray::Erase (CDoublePtrArray::iterator it) [virtual]`

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `brat::CDoublePtrArray::Erase()`.

5.5.4.21 `bool brat::CDoublePtrArray::Erase (int32_t index) [virtual]`

Delete an element referenced by index

Returns

true if no error, otherwise false

References `brat::CDoublePtrArray::Erase()`.

5.5.4.22 `bool brat::CArrayStringMap::Erase (CArrayStringMap::iterator it) [virtual]`

Delete an element referenced by it

Returns

true if no error, otherwise false

5.5.4.23 `bool brathl::CArrayStringMap::Erase (const string & key) [virtual]`

Delete an element by its key

Returns

true if no error, otherwise false

5.5.4.24 `bool brathl::CObArray::Erase (CBratObject * ob)`

Delete an element referenced by ob

Returns

true if no error, otherwise false

Referenced by `brathl::CObArray::Erase()`.

5.5.4.25 `bool brathl::CObArray::Erase (CObArray::iterator it) [virtual]`

Delete an element referenced by it

Returns

true if no error, otherwise false

5.5.4.26 `bool brathl::CObArray::Erase (int32_t index) [virtual]`

Delete an element referenced by index

Returns

true if no error, otherwise false

References `brathl::CObArray::Erase()`.

5.5.4.27 `bool brathl::CStringMap::Erase (CStringMap::iterator it) [virtual]`

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `brathl::CStringMap::Erase()`.

5.5.4.28 `bool brathl::CStringMap::Erase (const string & key) [virtual]`

Delete an element by its key

Returns

true if no error, otherwise false

References `brathl::CStringMap::Erase()`.

5.5.4.29 `bool brathl::CIntMap::Erase (CIntMap::iterator it)` [virtual]

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `brathl::CIntMap::Erase()`.

5.5.4.30 `bool brathl::CIntMap::Erase (const string & key)` [virtual]

Delete an element by its key

Returns

true if no error, otherwise false

References `brathl::CIntMap::Erase()`.

5.5.4.31 `bool brathl::CUIntMap::Erase (CUIntMap::iterator it)` [virtual]

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `brathl::CUIntMap::Erase()`.

5.5.4.32 `bool brathl::CUIntMap::Erase (const string & key)` [virtual]

Delete an element by its key

Returns

true if no error, otherwise false

References `brathl::CUIntMap::Erase()`.

5.5.4.33 `bool brathl::CDoubleMap::Erase (CDoubleMap::iterator it)` [virtual]

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `brathl::CDoubleMap::Erase()`.

5.5.4.34 `bool brathl::CDoubleMap::Erase (const string & key)` [virtual]

Delete an element by its key

Returns

true if no error, otherwise false

References `brathl::CDoubleMap::Erase()`.

5.5.4.35 `bool brathl::CObMap::Erase (CObMap::iterator it)` [virtual]

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `brathl::CObMap::Erase()`, and `brathl::CDataSet::EraseFieldSet()`.

5.5.4.36 `bool brathl::CObMap::Erase (const string & key)` [virtual]

Delete an element by its key

Returns

true if no error, otherwise false

References `brathl::CObMap::Erase()`.

5.5.4.37 `bool brathl::CObIntMap::Erase (CObIntMap::iterator it)` [virtual]

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `brathl::CObIntMap::Erase()`.

5.5.4.38 `bool brathl::CObIntMap::Erase (int32_t key)` [virtual]

Delete an element by its key

Returns

true if no error, otherwise false

References `brathl::CObIntMap::Erase()`.

5.5.4.39 `bool brathl::CObDoubleMap::Erase (CObDoubleMap::iterator it)` [virtual]

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `brathl::CObDoubleMap::Erase()`.

5.5.4.40 `bool brathl::CObDoubleMap::Erase (double key)` [virtual]

Delete an element by its key

Returns

true if no error, otherwise false

References `brathl::CObDoubleMap::Erase()`.

5.5.4.41 `bool brathl::CDoublePtrDoubleMap::Erase (CDoublePtrDoubleMap::iterator it)` [virtual]

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `brathl::CDoublePtrDoubleMap::Erase()`.

5.5.4.42 `bool brathl::CDoublePtrDoubleMap::Erase (double key)` [virtual]

Delete an element by its key

Returns

true if no error, otherwise false

References `brathl::CDoublePtrDoubleMap::Erase()`.

5.5.4.43 `bool brathl::CPtrMap::Erase (CPtrMap::iterator it)` [virtual]

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `brathl::CPtrMap::Erase()`.

5.5.4.44 `bool brathl::CPtrMap::Erase (const string & key)` [virtual]

Delete an element by its key

Returns

true if no error, otherwise false

References `brathl::CPtrMap::Erase()`.

5.5.4.45 `const CStringArray * brathl::CArrayStringMap::Exists (const string & key) const` [virtual]

Tests if an element identify by 'key' already exists

Returns

a string array value corresponding to the key; if exists, otherwise empty string

5.5.4.46 `string brathl::CStringMap::Exists (const string & key) const` [virtual]

Tests if an element identify by 'key' already exists

Returns

a string value corresponding to the key; if exists, otherwise empty string

5.5.4.47 `int32_t brathl::CIntMap::Exists (const string & key) const` [virtual]

Tests if an element identify by 'key' already exists

Returns

a integer value corresponding to the key; if exists, otherwise default value **CTools::m_defaultValueINT32** (p. 326)

References bratl::CTools::m_defaultValueINT32.

Referenced by bratl::CIntMap::operator[]().

5.5.4.48 `uint32_t bratl::CUIntMap::Exists (const string & key) const` [virtual]

Tests if an element identify by 'key' already exists

Returns

a integer value corresponding to the key; if exists, otherwise default value **CTools::m_defaultValueUINT32** (p. 326)

References bratl::CTools::m_defaultValueUINT32.

Referenced by bratl::CUIntMap::operator[]().

5.5.4.49 `double bratl::CDoubleMap::Exists (const string & key) const` [virtual]

Tests if an element identify by 'key' already exists

Returns

a double value corresponding to the key; if exists, otherwise default value **CTools::m_defaultValueDOUBLE** (p. 326)

References bratl::CTools::m_defaultValueDOUBLE.

Referenced by bratl::CDoubleMap::operator[]().

5.5.4.50 `CBratObject * bratl::COBMap::Exists (const string & key) const` [virtual]

Tests if an element identify by 'key' already exists

Returns

a CBratObject pointer if exists, otherwise NULL

5.5.4.51 `CBratObject * bratl::COBIntMap::Exists (int32_t key) const` [virtual]

Tests if an element identify by 'key' already exists

Returns

a CBratObject pointer if exists, otherwise NULL

5.5.4.52 `CBratObject * bratl::COBDoubleMap::Exists (double key) const` [virtual]

Tests if an element identify by 'key' already exists

Returns

a CBratObject pointer if exists, otherwise NULL

5.5.4.53 `DoublePtr * bratl::CDoublePtrDoubleMap::Exists (double key) const` [virtual]

Tests if an element identify by 'key' already exists

Returns

a CBratObject pointer if exists, otherwise NULL

5.5.4.54 `void * bratl::CPtrMap::Exists (const string & key) const` [virtual]

Tests if an element identify by 'key' already exists

Returns

a pointer if exists, otherwise NULL

5.5.4.55 `void bratl::CStringMap::GetKeys (CStringArray & keys, bool bRemoveAll=true) const` [virtual]

Gets keys of the map

Parameters

<i>keys</i>	[out] : the keys of the map
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

5.5.4.56 `void bratl::CUIntMap::GetKeys (CStringArray & keys, bool bRemoveAll=true)` [virtual]

Gets keys of the map

Parameters

<i>keys</i>	[out] : the keys of the map
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

5.5.4.57 `void bratl::CObMap::GetKeys (CStringArray & keys, bool bRemoveAll=true, bool bUnique=false)`
[virtual]

Gets keys of the map

Parameters

<i>keys</i>	[out] : the keys of the map
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

5.5.4.58 `void bratl::CObMap::GetKeys (CStringList & keys, bool bRemoveAll=true, bool bUnique=false)`
[virtual]

Gets keys of the map

Parameters

<i>keys</i>	[out] : the keys of the map
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

5.5.4.59 `void bratl::COblntMap::GetKeys (CIntArray & keys, bool bRemoveAll=true)` [virtual]

Gets keys of the map

Parameters

<i>keys</i>	[out] : the keys of the map
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

5.5.4.60 `void brathl::CObDoubleMap::GetKeys (CDoubleArray & keys, bool bRemoveAll =true) [virtual]`

Gets keys of the map

Parameters

<i>keys</i>	[out] : the keys of the map
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

References `brathl::CDoubleArray::Insert()`.

5.5.4.61 `void brathl::CDoublePtrDoubleMap::GetKeys (CDoubleArray & keys, bool bRemoveAll =true) [virtual]`

Gets keys of the map

Parameters

<i>keys</i>	[out] : the keys of the map
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

References `brathl::CDoubleArray::Insert()`.

5.5.4.62 `void brathl::CFloatArray::Insert (float * data, int32_t size) [virtual]`

Inserts an array of float at the end of the array

Parameters

<i>data</i>	[in] : array to be copied
<i>size</i>	[in] : array size to be copied

Referenced by `brathl::CFloatArray::operator=()`.

5.5.4.63 `void brathl::CFloatArray::Insert (const CFloatArray & vect, bool bEnd =true) [virtual]`

Inserts a **CFloatArray** (p. 248)

Parameters

<i>vect</i>	[in] : array to be copied
<i>bEnd</i>	[in] : insert values at the end if true, at the beginning if false

5.5.4.64 `void brathl::CFloatArray::Insert (const CFloatArray & vect, int32_t first, int32_t last, bool bEnd =true) [virtual]`

Inserts a partial **CFloatArray** (p. 248)

Parameters

<i>vect</i>	[in] : array to be copied
<i>first</i>	[in] : the position of the first element in the range of elements to be copied.
<i>last</i>	[in] : the position of the first element beyond the range of elements to be copied.
<i>bEnd</i>	[in] : insert values at the end if true, at the beginning if false

5.5.4.65 `void brathl::CDoubleArray::Insert (double * data, int32_t size) [virtual]`

Inserts an array of double at the end of the array

Parameters

<i>data</i>	[in] : array to be copied
<i>size</i>	[in] : array size to be copied

Referenced by `brathl::CObDoubleMap::GetKeys()`, `brathl::CDoublePtrDoubleMap::GetKeys()`, and `brathl::CDoubleArray::operator=()`.

5.5.4.66 `void brathl::CDoubleArray::Insert (const CDoubleArray & vect, bool bEnd = true) [virtual]`

Inserts a **CDoubleArray** (p. 195)

Parameters

<i>vect</i>	[in] : array to be copied
<i>bEnd</i>	[in] : insert values at the end if true, at the beginning if false

5.5.4.67 `void brathl::CDoubleArray::Insert (const CDoubleArray & vect, int32_t first, int32_t last, bool bEnd = true) [virtual]`

Inserts a partial **CDoubleArray** (p. 195)

Parameters

<i>vect</i>	[in] : array to be copied
<i>first</i>	[in] : the position of the first element in the range of elements to be copied.
<i>last</i>	[in] : the position of the first element beyond the range of elements to be copied.
<i>bEnd</i>	[in] : insert values at the end if true, at the beginning if false

5.5.4.68 `CStringArray * brathl::CArrayStringMap::Insert (const string & key, const CStringArray & str, bool withExcept = true) [virtual]`

Inserts a string

Parameters

<i>key</i>	: map key
<i>str</i>	: string value

Returns

the inserted string value or existing string value if key exists

References `BRATHL_LOGIC_ERROR`.

5.5.4.69 `string brathl::CStringMap::Insert (const string & key, const string & str, bool withExcept = true) [virtual]`

Inserts a string

Parameters

<i>key</i>	: map key
<i>str</i>	: string value

Returns

the inserted string value or existing string value if key exists

References BRATHL_LOGIC_ERROR.

Referenced by brathl::CStringMap::Insert().

5.5.4.70 void brathl::CStringMap::Insert (const CStringMap & *strmap*, bool *withExcept* = true) [virtual]

Inserts a string map

Parameters

<i>strmap</i>	: map to insert
<i>withExcept</i>	: true for exception handling, flse otherwise

Returns

the inserted string value or existing string value if key exists

References brathl::CStringMap::Insert().

5.5.4.71 int32_t brathl::CIntMap::Insert (const string & *key*, int32_t *value*, bool *withExcept* = true) [virtual]

Inserts an integer

Parameters

<i>key</i>	: map key
<i>value</i>	: int value

Returns

the inserted integer value or existing integer value if key exists

References BRATHL_LOGIC_ERROR.

Referenced by brathl::CIntMap::Insert().

5.5.4.72 void brathl::CIntMap::Insert (const CIntMap & *m*, bool *bRemoveAll* = true, bool *withExcept* = true) [virtual]

Inserts a **CIntMap** (p. 257)

Parameters

<i>map</i>	[in]: map
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

References brathl::CIntMap::Insert(), and brathl::CIntMap::RemoveAll().

5.5.4.73 uint32_t brathl::CUIntMap::Insert (const string & *key*, uint32_t *value*, bool *withExcept* = true) [virtual]

Inserts an integer

Parameters

<i>key</i>	: map key
<i>value</i>	: int value

Returns

the inserted integer value or existing unsigned integer value if key exists

References BRATHL_LOGIC_ERROR.

Referenced by brathl::CUIntMap::Insert().

5.5.4.74 `void brathl::CUIntMap::Insert (const CUIntMap & m, bool bRemoveAll = true, bool withExcept = true) [virtual]`

Inserts a **CUIntMap** (p. 352)

Parameters

<i>map</i>	[in]: map
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

References brathl::CUIntMap::Insert(), and brathl::CUIntMap::RemoveAll().

5.5.4.75 `void brathl::CUIntMap::Insert (const CStringArray & keys, uint32_t initValue, bool bRemoveAll = true, bool withExcept = true) [virtual]`

Inserts a CStringArray as keys and initial value

Parameters

<i>keys</i>	[in]: map keys to insert
<i>initValue</i>	[in]: value of the keys
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

References brathl::CUIntMap::Insert(), and brathl::CUIntMap::RemoveAll().

5.5.4.76 `void brathl::CUIntMap::Insert (const CStringArray & keys, const CUIntArray & values, bool bRemoveAll = true, bool withExcept = true) [virtual]`

Inserts a CStringArray as keys and a **CUIntArray** (p. 351) as value

Parameters

<i>keys</i>	[in]: keys to insert
<i>values</i>	[in]: values to insert
<i>bRemoveAll</i>	[in] : if true, remove keys array element before filling the keys

References BRATHL_LOGIC_ERROR, brathl::CTools::Format(), brathl::CUIntMap::Insert(), and brathl::CUIntMap::RemoveAll().

5.5.4.77 `double brathl::CDoubleMap::Insert (const string & key, double value, bool withExcept = true) [virtual]`

Inserts an double

Parameters

<i>key</i>	: map key
<i>value</i>	: double value

Returns

the inserted double value or existing double value if key exists

References BRATHL_LOGIC_ERROR.

5.5.4.78 `CBratObject * brathl::CObMap::Insert (const string & key, CBratObject * ob, bool withExcept = true)`
`[virtual]`

Inserts a CBratObject object

Parameters

<i>key</i>	: CBratObject name (map key)
<i>value</i>	: CBratObject value
<i>withExcept</i>	: true for exception handling, flse otherwise

Returns

CBratObject object or NULL if error

References BRATHL_LOGIC_ERROR.

Referenced by `brathl::CObMap::Insert()`, `brathl::CDataSet::InsertFieldSet()`, and `brathl::CObMap::RenameKey()`.

5.5.4.79 `void brathl::CObMap::Insert (const CObMap & obMap, bool withExcept = true)` `[virtual]`

Inserts a **CObMap** (p. 274)

Parameters

<i>obMap</i>	: CObMap (p. 274) to insert
<i>withExcept</i>	: true for exception handling, flse otherwise

References `brathl::CObMap::Insert()`.

5.5.4.80 `CBratObject * brathl::COblntMap::Insert (int32_t key, CBratObject * ob, bool withExcept = true)` `[virtual]`

Inserts a CBratObject object

Parameters

<i>key</i>	: CBratObject name (map key)
<i>value</i>	: CBratObject value
<i>withExcept</i>	: true for exception handling, flse otherwise

Returns

CBratObject object or NULL if error

References BRATHL_LOGIC_ERROR.

Referenced by `brathl::COblntMap::Insert()`, and `brathl::COblntMap::RenameKey()`.

5.5.4.81 `void brathl::COblntMap::Insert (const COblntMap & obMap, bool withExcept = true)` `[virtual]`

Inserts a **COblntMap** (p. 271)

Parameters

<i>obMap</i>	: CObMap (p. 274) to insert
<i>withExcept</i>	: true for exception handling, flse otherwise

References `brathl::COblntMap::Insert()`.

5.5.4.82 `CBratObject * brathl::CObDoubleMap::Insert (double key, CBratObject * ob, bool withExcept = true)`
[virtual]

Inserts a CBratObject object

Parameters

<i>key</i>	: CBratObject name (map key)
<i>value</i>	: CBratObject value
<i>withExcept</i>	: true for exception handling, flse otherwise

Returns

CBratObject object or NULL if error

References BRATHL_LOGIC_ERROR.

Referenced by brathl::CObDoubleMap::Insert(), and brathl::CObDoubleMap::RenameKey().

5.5.4.83 `void brathl::CObDoubleMap::Insert (const CObDoubleMap & obMap, bool withExcept = true)` [virtual]

Inserts a **CObDoubleMap** (p. 271)

Parameters

<i>obMap</i>	: CObMap (p. 274) to insert
<i>withExcept</i>	: true for exception handling, flse otherwise

References brathl::CObDoubleMap::Insert().

5.5.4.84 `DoublePtr * brathl::CDoublePtrDoubleMap::Insert (double key, DoublePtr * ob, bool withExcept = true)`
[virtual]

Inserts a DoublePtr* object

Parameters

<i>key</i>	: DoublePtr* name (map key)
<i>value</i>	: DoublePtr* value
<i>withExcept</i>	: true for exception handling, flse otherwise

Returns

DoublePtr* object or NULL if error

References BRATHL_LOGIC_ERROR.

Referenced by brathl::CDoublePtrDoubleMap::RenameKey().

5.5.4.85 `void * brathl::CPtrMap::Insert (const string & key, void * ptr, bool withExcept = true)` [virtual]

Inserts a pointer

Parameters

<i>key</i>	: keymap
<i>value</i>	: pointer value
<i>withExcept</i>	: true for exception handling, flse otherwise

Returns

pointer or NULL if error

References BRATHL_LOGIC_ERROR.

Referenced by brathl::CPtrMap::Insert().

5.5.4.86 void brathl::CPtrMap::Insert (const CPtrMap & ptrMap, bool withExcept = true) [virtual]

Inserts a **CPtrMap** (p. 315)

Parameters

<i>obMap</i>	: CPtrMap (p. 315) to insert
<i>withExcept</i>	: true for exception handling, flse otherwise

References brathl::CPtrMap::Insert().

5.5.4.87 string brathl::CStringMap::IsValue (const string & value) [virtual]

Tests if an element value exists

Returns

a string key corresponding to the value (or the first key found, if some values are the same); if exists, otherwise empty string

5.5.4.88 virtual bool brathl::CStringArray::operator!= (const CStringArray & vect) [inline],[virtual]

Inequality operator overload Array are unequal if they are not equal

5.5.4.89 virtual bool brathl::CUIntArray::operator!= (const CUIntArray & vect) [inline],[virtual]

Inequality operator overload Array are unequal if they are not equal

5.5.4.90 virtual bool brathl::CUInt64Array::operator!= (const CUInt64Array & vect) [inline],[virtual]

Inequality operator overload Array are unequal if they are not equal

5.5.4.91 virtual bool brathl::CUInt16Array::operator!= (const CUInt16Array & vect) [inline],[virtual]

Inequality operator overload Array are unequal if they are not equal

5.5.4.92 virtual bool brathl::CUInt8Array::operator!= (const CUInt8Array & vect) [inline],[virtual]

Inequality operator overload Array are unequal if they are not equal

5.5.4.93 virtual bool brathl::CDoubleArray::operator!= (const CDoubleArray & vect) [inline],[virtual]

Inequality operator overload Array are unequal if they are not equal

5.5.4.94 const CStringList & brathl::CStringList::operator= (const CStringList & lst) [virtual]

Copy a new **CStringList** (p. 319) to the object

Referenced by brathl::CProductList::Set().

5.5.4.95 const CIntList & brathl::CIntList::operator= (const CIntList & lst)

Copy a new **CIntList** (p. 257) to the object

5.5.4.96 `const CObList & bratl::CObList::operator= (const CObList & lst) [virtual]`

Copy a new **CStringList** (p. 319) to the object

References `bratl::CObList::RemoveAll()`.

5.5.4.97 `const CStringArray & bratl::CStringArray::operator= (const CStringArray & vect) [virtual]`

Copy a new **CStringArray** to the object

5.5.4.98 `const CIntArray & bratl::CIntArray::operator= (const CIntArray & vect) [virtual]`

Copy a new **CIntArray** (p. 251) to the object

5.5.4.99 `const CInt64Array & bratl::CInt64Array::operator= (const CInt64Array & vect) [virtual]`

Copy a new **CInt64Array** (p. 250) to the object

5.5.4.100 `const CUIntArray & bratl::CUIntArray::operator= (const CUIntArray & vect) [virtual]`

Copy a new **CUIntArray** (p. 351) to the object

5.5.4.101 `const CUInt64Array & bratl::CUInt64Array::operator= (const CUInt64Array & vect) [virtual]`

Copy a new **CUIntArray** (p. 351) to the object

5.5.4.102 `const CInt16Array & bratl::CInt16Array::operator= (const CInt16Array & vect) [virtual]`

Copy a new **CInt16Array** (p. 249) to the object

5.5.4.103 `const CUInt16Array & bratl::CUInt16Array::operator= (const CUInt16Array & vect) [virtual]`

Copy a new **CUInt16Array** (p. 348) to the object

5.5.4.104 `const CInt8Array & bratl::CInt8Array::operator= (const CInt8Array & vect) [virtual]`

Copy a new **CInt8Array** (p. 251) to the object

5.5.4.105 `const CUInt8Array & bratl::CUInt8Array::operator= (const CUInt8Array & vect) [virtual]`

Copy a new **CUInt8Array** (p. 350) to the object

5.5.4.106 `const CFloatArray & bratl::CFloatArray::operator= (const CFloatArray & vect) [virtual]`

Copy a new **CFloatArray** (p. 248) to the object

References `bratl::CFloatArray::Insert()`.

5.5.4.107 `const CDoubleArray & bratl::CDoubleArray::operator= (const CDoubleArray & vect) [virtual]`

Copy a new **CDoubleArray** (p. 195) to the object

References `bratl::CDoubleArray::Insert()`.

5.5.4.108 `const CObArray & bratl::CObArray::operator= (const CObArray & lst) [virtual]`

Copy a new **CObArray** (p. 270) to the object

References `bratl::CObArray::RemoveAll()`.

5.5.4.109 `bool bratl::CStringArray::operator== (const CStringArray & vect) [virtual]`

Equality operator overload Array are equal if they have same size and the same element values (at the same position)

5.5.4.110 `bool brathl::CIntArray::operator==(const CIntArray & vect) [virtual]`

Equality operator overload Array are equal if they have same size and the same element values (at the same position)

5.5.4.111 `bool brathl::CInt64Array::operator==(const CInt64Array & vect) [virtual]`

Equality operator overload Array are equal if they have same size and the same element values (at the same position)

5.5.4.112 `bool brathl::CUIntArray::operator==(const CUIntArray & vect) [virtual]`

Equality operator overload Array are equal if they have same size and the same element values (at the same position)

5.5.4.113 `bool brathl::CUInt64Array::operator==(const CUInt64Array & vect) [virtual]`

Equality operator overload Array are equal if they have same size and the same element values (at the same position)

5.5.4.114 `bool brathl::CUInt16Array::operator==(const CUInt16Array & vect) [virtual]`

Equality operator overload Array are equal if they have same size and the same element values (at the same position)

5.5.4.115 `bool brathl::CUInt8Array::operator==(const CUInt8Array & vect) [virtual]`

Equality operator overload Array are equal if they have same size and the same element values (at the same position)

5.5.4.116 `bool brathl::CDoubleArray::operator==(const CDoubleArray & vect) [virtual]`

Equality operator overload Array are equal if they have same size and the same element values (at the same position)

5.5.4.117 `int32_t brathl::CIntMap::operator[] (const string & key) [virtual]`

`operator[]` redefinition. Searches an integer value identify by 'key'.

Parameters

<code>key</code>	: string keyword
------------------	------------------

Returns

the interger value if found, default value **CTools::m_defaultValueINT32** (p. 326) if not found

References `brathl::CIntMap::Exists()`.

5.5.4.118 `uint32_t brathl::CUIntMap::operator[] (const string & key) [virtual]`

`operator[]` redefinition. Searches an integer value identify by 'key'.

Parameters

<code>key</code>	: string keyword
------------------	------------------

Returns

the interger value if found, default value **CTools::m_defaultValueUINT32** (p. 326) if not found

References bratl::CUIntMap::Exists().

5.5.4.119 `double bratl::CDoubleMap::operator[] (const string & key) [virtual]`

`operator[]` redefinition. Searches an integer value identifiy by 'key'.

Parameters

<i>key</i>	: string keyword
------------	------------------

Returns

the double value if found, default value **CTools::m_defaultValueDOUBLE** (p. 326) if not found

References bratl::CDoubleMap::Exists().

5.5.4.120 `CBratObject * bratl::CObMap::operator[] (const string & key) [virtual]`

`operator[]` redefinition. Searches a CBratObject object identifiy by 'key'. DON'T USE this syntax if you are not sure the key exists, there's a bug in STL, after calling 'record = m_recordSetMap[recordSetName]', if key not existed and the map is empty then the key exists in the map and points to a NULL object CBratObject *o = myMap[key] -> use Exists method instead ;

Parameters

<i>key</i>	: CBratObject keyword
------------	-----------------------

Returns

a pointer to the CBratObject object if found, NULL if not found

5.5.4.121 `CBratObject * bratl::CObIntMap::operator[] (int32_t key) [virtual]`

`operator[]` redefinition. Searches a CBratObject object identifiy by 'key'. DON'T USE this syntax if you are not sure the key exists, there's a bug in STL, after calling 'record = m_recordSetMap[recordSetName]', if key not existed and the map is empty then the key exists in the map and points to a NULL object CBratObject *o = myMap[key] -> use Exists method instead ;

Parameters

<i>key</i>	: CBratObject keyword
------------	-----------------------

Returns

a pointer to the CBratObject object if found, NULL if not found

5.5.4.122 `CBratObject * bratl::CObDoubleMap::operator[] (double key) [virtual]`

`operator[]` redefinition. Searches a CBratObject object identifiy by 'key'. DON'T USE this syntax if you are not sure the key exists, there's a bug in STL, after calling 'record = m_recordSetMap[recordSetName]', if key not existed and the map is empty then the key exists in the map and points to a NULL object CBratObject *o = myMap[key] -> use Exists method instead ;

Parameters

<i>key</i>	: CBratObject keyword
------------	-----------------------

Returns

a pointer to the CBratObject object if found, NULL if not found

5.5.4.123 DoublePtr * bratl::CDoublePtrDoubleMap::operator[] (double key) [virtual]

operator[] redefinition. Searches a CBratObject object identify by 'key'. DON'T USE this syntax if you are not sure the key exists, there's a bug in STL, after calling 'record = m_recordSetMap[recordSetName]', if key not existed and the map is empty then the key exists in the map and points to a NULL object CBratObject *o = myMap[key] -> use Exists method instead ;

Parameters

<i>key</i>	: CBratObject keyword
------------	-----------------------

Returns

a pointer to the CBratObject object if found, NULL if not found

5.5.4.124 void * bratl::CPtrMap::operator[] (const string & key) [virtual]

operator[] redefinition. Searches a CBratObject object identify by 'key'. DON'T USE this syntax if you are not sure the key exists, there's a bug in STL, after calling 'record = m_recordSetMap[recordSetName]', if key not existed and the map is empty then the key exists in the map and points to a NULL object void *p = myMap[key] -> use Exists method instead ;

Parameters

<i>key</i>	: CBratObject keyword
------------	-----------------------

Returns

a pointer to the pointer if found, NULL if not found

5.5.4.125 void bratl::CObList::RemoveAll () [virtual]

Remove all elements and clear the list

Reimplemented in **bratl::CField::CListField** (p. 259).

Referenced by bratl::CObList::operator=(), bratl::CField::CListField::RemoveAll(), and bratl::CObList::~~CObList().

5.5.4.126 void bratl::CStringArray::RemoveAll () [virtual]

Remove all elements and clear the list

5.5.4.127 void bratl::CFloatArray::RemoveAll () [virtual]

Remove all elements and clear the list

5.5.4.128 void bratl::CDoubleArray::RemoveAll () [virtual]

Remove all elements and clear the list

5.5.4.129 void brathl::CDoublePtrArray::RemoveAll () [virtual]

Remove all elements and clear the list

Referenced by brathl::CDoublePtrArray::~~CDoublePtrArray().

5.5.4.130 void brathl::CArrayDoublePtrArray::RemoveAll () [virtual]

Remove all elements and clear the list

5.5.4.131 void brathl::CArrayDoubleArray::RemoveAll () [virtual]

Remove all elements and clear the list

5.5.4.132 void brathl::CArrayStringMap::RemoveAll () [virtual]

Remove all elements and clear the map

5.5.4.133 void brathl::CObStack::RemoveAll () [virtual]

Remove all elements and clear the list

References brathl::CObStack::m_bDelete.

Referenced by brathl::CObStack::~~CObStack().

5.5.4.134 void brathl::CObArray::RemoveAll () [virtual]

Remove all elements and clear the list

Reimplemented in **brathl::CDataSet** (p. 173).

Referenced by brathl::CObArray::operator=(), and brathl::CObArray::~~CObArray().

5.5.4.135 void brathl::CStringMap::RemoveAll () [virtual]

Remove all elements and clear the map

Referenced by brathl::CStringMap::~~CStringMap().

5.5.4.136 void brathl::CIntMap::RemoveAll () [virtual]

Remove all elements and clear the map

Referenced by brathl::CIntMap::Insert(), and brathl::CIntMap::~~CIntMap().

5.5.4.137 void brathl::CUIntMap::RemoveAll () [virtual]

Remove all elements and clear the map

Referenced by brathl::CUIntMap::Insert(), and brathl::CUIntMap::~~CUIntMap().

5.5.4.138 void brathl::CDoubleMap::RemoveAll () [virtual]

Remove all elements and clear the map

Referenced by brathl::CDoubleMap::~~CDoubleMap().

5.5.4.139 void brathl::CObMap::RemoveAll () [virtual]

Remove all elements and clear the map

Referenced by brathl::CDataSet::RemoveAll(), and brathl::CObMap::~~CObMap().

5.5.4.140 void brathl::CObIntMap::RemoveAll () [virtual]

Remove all elements and clear the map

Referenced by `brathl::COblntMap::~~COblntMap()`.

5.5.4.141 `void brathl::CObDoubleMap::RemoveAll () [virtual]`

Remove all elements and clear the map

Referenced by `brathl::CObDoubleMap::~~CObDoubleMap()`.

5.5.4.142 `void brathl::CDoublePtrDoubleMap::RemoveAll () [virtual]`

Remove all elements and clear the map

Referenced by `brathl::CDoublePtrDoubleMap::~~CDoublePtrDoubleMap()`.

5.5.4.143 `void brathl::CPtrMap::RemoveAll () [virtual]`

Remove all elements and clear the map

Referenced by `brathl::CPtrMap::~~CPtrMap()`.

5.5.4.144 `bool brathl::CObMap::RenameKey (const string & oldKey, const string & newKey)`

Rename a key

Parameters

<i>oldKey</i>	: old key
<i>newKey</i>	: new key

Returns

true if key is renamed, otherwise false

References `brathl::CObMap::Insert()`.

5.5.4.145 `bool brathl::COblntMap::RenameKey (int32_t oldKey, int32_t newKey)`

Rename a key

Parameters

<i>oldKey</i>	: old key
<i>newKey</i>	: new key

Returns

true if key is renamed, otherwise false

References `brathl::COblntMap::Insert()`.

5.5.4.146 `bool brathl::CObDoubleMap::RenameKey (double oldKey, double newKey)`

Rename a key

Parameters

<i>oldKey</i>	: old key
<i>newKey</i>	: new key

Returns

true if key is renamed, otherwise false

References brathl::CObDoubleMap::Insert().

5.5.4.147 `bool brathl::CDoublePtrDoubleMap::RenameKey (double oldKey, double newKey)`

Rename a key

Parameters

<i>oldKey</i>	: old key
<i>newKey</i>	: new key

Returns

true if key is renamed, otherwise false

References brathl::CDoublePtrDoubleMap::Insert().

5.5.4.148 `void brathl::CArrayStringMap::Set (const CArrayStringMap & a) [virtual]`

Inserts a string map

Parameters

<i>strmap</i>	: map to insert
<i>withExcept</i>	: true for exception handling, flse otherwise

Returns

the inserted string value or existing string value if key exists

5.5.5 Variable Documentation

5.5.5.1 `const string brathl::UNKNOWN_PRODUCT_CLASS = "UNKNOWN"`

External files access.

Version

1.0

5.6 Criteria

Classes

- class **brathl::CCriteria**
- class **brathl::CCriteriaCycle**
- class **brathl::CCriteriaCycleInfo**
- class **brathl::CCriteriaDatetime**
- class **brathl::CCriteriaDatetimeInfo**
- class **brathl::CCriterialInfo**
- class **brathl::CCriteriaLatLon**
- class **brathl::CCriteriaLatLonInfo**
- class **brathl::CCriteriaPass**
- class **brathl::CCriteriaPassInfo**
- class **brathl::CCriteriaPassInt**
- class **brathl::CCriteriaPassIntInfo**
- class **brathl::CCriteriaPassString**
- class **brathl::CCriteriaPassStringInfo**
- class **brathl::CDataSet**
- class **brathl::CField**
- class **brathl::CFieldArray**
- class **brathl::CFieldBasic**
- class **brathl::CFieldIndexData**
- class **brathl::CFieldNetCdf**
- class **brathl::CFieldNetCdfCF**
- class **brathl::CFieldNetCdfCFAttr**
- class **brathl::CFieldRecord**
- class **brathl::CFieldSet**
- class **brathl::CFieldSetArrayDbI**
- class **brathl::CFieldSetDbI**
- class **brathl::CFieldSetString**
- class **brathl::CProduct::CInfo**
- class **brathl::CProduct::CListInfo**
- class **brathl::CMapProduct**
- class **brathl::CProductAop**
- class **brathl::CProductCryosat**
- class **brathl::CProductEnvisat**
- class **brathl::CProductErs**
- class **brathl::CProductErsWAP**
- class **brathl::CProductGfo**
- class **brathl::CProductJason**
- class **brathl::CProductJason2**
- class **brathl::CProductList**
- class **brathl::CProductNetCdf**
- class **brathl::CProductNetCdfCF**
- class **brathl::CProductPodaac**
- class **brathl::CProductRads**
- class **brathl::CProductRiverLake**
- class **brathl::CProductTopex**
- class **brathl::CProductTopexSDR**
- class **brathl::CRecord**
- class **brathl::CRecordSet**
- class **brathl::CTreeField**

Functions

- void **bratl::CProduct::AddCriteria** (bool force=false)
- void **bratl::CProduct::AddCriteria** (CCriteria *criteria, bool erase=true)
- void **bratl::CProduct::AddCriteria** (CProduct *product)
- void **bratl::CMapProduct::AddCriteriaToProducts** ()
- void **bratl::CProduct::AddFile** (const string &fileName, bool bEnd=true, bool checkFiles=true)
- void **bratl::CProduct::AddFile** (const CStringList &fileNameList, bool bEnd=true, bool checkFiles=true)
- virtual void **bratl::CProduct::AddInternalHighResolutionFieldCalculation** ()
- CInfo * **bratl::CProduct::CListInfo::AddNew** ()
- virtual void **bratl::CProduct::AddOffset** (double value, CField *field=NULL)
- bool **bratl::CProduct::AddRecordNameToField** (const CExpression &expr, const string &dataSetName, CExpression &exprOut, string &errorMsg)
- bool **bratl::CProduct::AddRecordNameToField** (const string &in, const string &dataSetName, string &out, string &errorMsg)
- bool **bratl::CProduct::AddRecordNameToField** (const string &in, const string &dataSetName, const C-StringArray &fieldsIn, string &out, string &errorMsg)
- bool **bratl::CProduct::AddRecordNameToField** (CProductAliases *productAliases, string &errorMsg)
- virtual void **bratl::CProduct::AddSameFieldName** (const string &fieldNameToSearch, CStringArray &arrayFieldsAdded)
- void **bratl::CCriteriaPassInt::Adjust** ()
- virtual void **bratl::CProduct::ApplyCriteria** (CStringList &filteredFileList, const string &logFileName="")
- virtual bool **bratl::CProduct::ApplyCriteriaCycle** (CCriterialInfo *criterialInfo)
- virtual bool **bratl::CProduct::ApplyCriteriaDatetime** (CCriterialInfo *criterialInfo)
- virtual bool **bratl::CProduct::ApplyCriteriaLatLon** (CCriterialInfo *criterialInfo)
- virtual bool **bratl::CProduct::ApplyCriteriaPass** (CCriterialInfo *criterialInfo)
- virtual bool **bratl::CProduct::ApplyCriteriaPassInt** (CCriterialInfo *criterialInfo)
- virtual bool **bratl::CProduct::ApplyCriteriaPassString** (CCriterialInfo *criterialInfo)
- CInfo * **bratl::CProduct::CListInfo::Back** (bool withExcept=true)
- void **bratl::CProduct::BuildCriteriaFieldsToRead** (CRecordDataMap &listRecord)
- **bratl::CCriteriaPass::CCriteriaPass** ()
Empty CCriteriaPass (p. 160) ctor.
- **bratl::CCriteriaPassInt::CCriteriaPassInt** ()
Empty CCriteriaPassInt (p. 164) ctor.
- **bratl::CCriteriaPassInt::CCriteriaPassInt** (CCriteriaPassInt &c)
- **bratl::CCriteriaPassInt::CCriteriaPassInt** (CCriteriaPassInt *c)
- **bratl::CCriteriaPassInt::CCriteriaPassInt** (int32_t from, int32_t to)
- **bratl::CCriteriaPassInt::CCriteriaPassInt** (const string &from, const string &to)
- **bratl::CCriteriaPassInt::CCriteriaPassInt** (const CStringArray &array)
- **bratl::CCriteriaPassString::CCriteriaPassString** ()
Empty CCriteriaPassString (p. 167) ctor.
- **bratl::CCriteriaPassString::CCriteriaPassString** (CCriteriaPassString &c)
- **bratl::CCriteriaPassString::CCriteriaPassString** (CCriteriaPassString *c)
- **bratl::CCriteriaPassString::CCriteriaPassString** (const string &passes, const string &delimiter=CCriteriaPassString::m_delimiter)
- **bratl::CCriteriaPassString::CCriteriaPassString** (const CStringArray &array)
- static bool **bratl::CProduct::CheckAliases** (const string &fileName, CStringArray &errors)
- bool **bratl::CProduct::CheckAliases** (CStringArray &errors)
- virtual void **bratl::CProduct::CheckConsistencyHighResolutionField** (CFieldSetArrayDbI *fieldSetArrayDbI)
- bool **bratl::CProduct::CheckFieldNames** (const CExpression &expr, const string &dataSetName, CStringArray &fieldNamesNotFound)
- bool **bratl::CProduct::CheckFieldNames** (const CExpression &expr, CStringArray &fieldNamesNotFound)
- bool **bratl::CProduct::CheckFieldNames** (const CStringArray *fieldNames, const string &dataSetName, CStringArray &fieldNamesNotFound)

- void **brathl::CProduct::CheckFields** (bool convertDate=false)
- bool **brathl::CProductList::CheckFileList** ()
- virtual void **brathl::CProduct::CheckFileOpened** ()
- bool **brathl::CProductList::CheckFiles** (bool onlyFirstFile=false)
- bool **brathl::CProduct::CheckFiles** ()
- bool **brathl::CProductList::CheckFilesNetCDF** ()
- virtual CProduct * **brathl::CProduct::Clone** ()
- virtual bool **brathl::CProduct::Close** ()
- **brathl::CMapProduct::CMapProduct** ()
CIntMap (p. 257) ctor.
- static void **brathl::CProduct::CodaInit** ()
- static void **brathl::CProduct::CodaRelease** ()
- static CProduct * **brathl::CProduct::Construct** (CStringArray &fileNameArray)
- static CProduct * **brathl::CProduct::Construct** (CStringList &fileNameList)
- static CProduct * **brathl::CProduct::Construct** (CProductList &fileNameList)
- static CProduct * **brathl::CProduct::Construct** (const string &fileName)
- void **brathl::CProduct::ConvertDate** (CDoubleArray &vect)
- **brathl::CProduct::CProduct** ()
Empty CProduct ctor.
- **brathl::CProduct::CProduct** (const string &fileName)
- **brathl::CProduct::CProduct** (const CStringList &fileNameList)
- **brathl::CProductGeneric::CProductGeneric** ()
Empty CProductGeneric ctor.
- **brathl::CProductGeneric::CProductGeneric** (const string &fileName)
- **brathl::CProductGeneric::CProductGeneric** (const CStringList &fileNameList)
- **brathl::CProductList::CProductList** ()
Empty CProductList (p. 298) ctor.
- **brathl::CProductList::CProductList** (const CProductList &p)
- **brathl::CProductList::CProductList** (const string &fileName)
- **brathl::CProductList::CProductList** (const CStringList &fileNameList)
- **brathl::CProductList::CProductList** (const CStringArray &fileNameArray)
- void **brathl::CProduct::CreateFieldIndexData** ()
- void **brathl::CProduct::CreateFieldIndexes** (CFieldArray *field)
- void **brathl::CProduct::CreateLogFile** (const string &logFileName, uint32_t mode=CFile::modeWrite|CFile::typeText)
- string **brathl::CProduct::DatasetRecordsNumberToString** (const CIntMap &datasetRecordsNumber)
- void **brathl::CProduct::DeleteLogFile** ()
- virtual void **brathl::CCriteriaPass::Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **brathl::CProductList::Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **brathl::CCriteriaPassString::Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **brathl::CCriteriaPassInt::Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **brathl::CProduct::Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **brathl::CMapProduct::Dump** (ostream &fOut=cerr)
- void **brathl::CProduct::DumpDictionary** (ostream &fOut=cout)
- void **brathl::CProduct::DumpDictionary** (const string &outputFileName)
- virtual void **brathl::CProduct::EndApplyCriteriaStats** (const CStringList &filteredFileList)
- void **brathl::CProduct::ExpandArray** ()
- void **brathl::CProduct::ExpandFieldsArray** ()

- virtual void **brathl::CProduct::ExtractDatasetNamesFromFields** (const **CStringList** &listFields, **CStringList** &datasetNames)
- static void **brathl::CCriteriaPassString::ExtractPass** (const string &passes, CStringArray &arrayPass, const string &delimiter=CCriteriaPassString::m_delimiter)
- static void **brathl::CCriteriaPassString::ExtractPass** (const CStringArray &array, CStringArray &arrayPass)
- virtual void **brathl::CProduct::FillDescription** ()
- void **brathl::CProduct::FillListFields** (const string &key)
- **CField** * **brathl::CProduct::FindFieldByInternalName** (const string &internalFieldName, bool withExcept=true)
- **CField** * **brathl::CProduct::FindFieldByName** (const string &fieldName, const string &dataSetName, bool withExcept=true, string *errorMsg=NULL, bool showTrace=true)
- **CField** * **brathl::CProduct::FindFieldByName** (const string &fieldName, bool withExcept=true, string *errorMsg=NULL, bool showTrace=true)
- virtual bool **brathl::CProduct::FindParentToRead** (**CField** *fromField, **CObList** *parentFieldList)
- **CInfo** * **brathl::CProduct::CListInfo::Front** (bool withExcept=true)
- const **CProductAlias** * **brathl::CProduct::GetAlias** (const string &key)
- const **CProductAliases** * **brathl::CProduct::GetAliases** ()
- const **CStringMap** * **brathl::CProduct::GetAliasesAsString** () const
- static const **CStringMap** * **brathl::CProduct::GetAliasesAsString** (const **CProduct** *product)
- string **brathl::CProduct::GetAliasExpandedValue** (const string &key)
- void **brathl::CProduct::GetAliasKeys** (CStringArray &keys)
- string **brathl::CCriteriaPassString::GetAsText** (const string &delimiter=CCriteriaPassString::m_delimiter)
- string **brathl::CCriteriaPassInt::GetAsText** (const string &delimiter=CCriteriaPassInt::m_delimiter)
- bool **brathl::CProduct::GetCreateVirtualField** ()
- static **CCriteriaPass** * **brathl::CCriteriaPass::GetCriteria** (**CBratObject** *ob, bool withExcept=true)
- static **CCriteriaPassString** * **brathl::CCriteriaPassString::GetCriteria** (**CBratObject** *ob, bool withExcept=true)
- **CCriteria** * **brathl::CProduct::GetCriteria** (**CCriterialInfo** *criterialInfo)
- static **CCriteriaPassInt** * **brathl::CCriteriaPassInt::GetCriteria** (**CBratObject** *ob, bool withExcept=true)
- virtual string **brathl::CProduct::GetCurrentFileName** ()
- virtual int32_t **brathl::CProduct::GetCurrentRecordNumber** ()
- **CCriteriaCycle** * **brathl::CProduct::GetCycleCriteria** ()
- **CCriteriaCycleInfo** * **brathl::CProduct::GetCycleCriterialInfo** ()
- **CStringArray** * **brathl::CProduct::GetDataDictionaryFieldNames** (bool forceReload=false)
- **CStringArray** * **brathl::CProduct::GetDataDictionaryFieldNamesWithDatasetName** (bool forceReload=false)
- **CDataSet** * **brathl::CProduct::GetDataSet** ()
- string **brathl::CProduct::GetDataSetNameToRead** ()
- virtual bool **brathl::CProduct::GetDateMinMax** (**CDatePeriod** &datePeriodMinMax)
- virtual bool **brathl::CProduct::GetDateMinMax** (**CDate** &dateMin, **CDate** &dateMax)
- **CCriteriaDatetime** * **brathl::CProduct::GetDatetimeCriteria** ()
- **CCriteriaDatetimelInfo** * **brathl::CProduct::GetDatetimeCriterialInfo** ()
- const string & **brathl::CProduct::GetDescription** ()
- bool **brathl::CProduct::GetDisableTrace** ()
- bool **brathl::CProduct::GetExpandArray** ()
- string **brathl::CProduct::GetFieldSpecificUnit** (const string &key)
- **CStringMap** * **brathl::CProduct::GetFieldSpecificUnits** ()
- **CStringArray** * **brathl::CProduct::GetFieldToTranspose** ()
- double **brathl::CProduct::GetForceLatMaxCriteriaValue** ()
- double **brathl::CProduct::GetForceLatMinCriteriaValue** ()
- virtual bool **brathl::CProduct::GetForceReadDataOneByOne** ()
- int32_t **brathl::CCriteriaPassInt::GetFrom** ()
- int32_t **brathl::CProduct::GetIndexProcessedFile** ()
- bool **brathl::CProduct::GetInfoArray** ()
- bool **brathl::CProduct::GetInfoRecord** (int32_t nbDims=1, const long dim[]=DEFAULT_DIM)

- `bool brathl::CProduct::GetInfoSpecial (int32_t nbDims=1, const long dim[]=DEFAULT_DIM)`
- `static CMapProduct & brathl::CMapProduct::GetInstance ()`
- `virtual string brathl::CProduct::GetLabel ()`
- `virtual string brathl::CProduct::GetLatitudeFieldName ()`
- `CCriteriaLatLon * brathl::CProduct::GetLatLonCriteria ()`
- `CCriteriaLatLonInfo * brathl::CProduct::GetLatLonCriteriaInfo ()`
- `virtual bool brathl::CProduct::GetLatLonMinMax (double &latMin, double &lonMin, double &latMax, double &lonMax)`
- `virtual bool brathl::CProduct::GetLatLonMinMax (CLatLonRect &latlonRectMinMax)`
- `CStringList * brathl::CProduct::GetListFieldOrigin ()`
- `virtual string brathl::CProduct::GetLongitudeFieldName ()`
- `const string brathl::CProductList::GetMessage ()`
- `virtual void brathl::CProduct::GetMinMaxNumberOfRecords (int32_t &min, int32_t &max, CIntMap *datasetRecordsNumber=NULL, int32_t minThreshold=-1)`
- `void brathl::CProduct::GetNamesCaseSensitive (const CStringArray &fieldsIn, CStringArray &fieldsOut-NoCaseSensitive, CStringArray &fieldsOutCaseSensitive, bool forceReload=false)`
- `virtual int32_t brathl::CProduct::GetNumberOfRecords ()`
- `virtual int32_t brathl::CProduct::GetNumberOfRecords (const string &dataSetName)`
- `virtual void brathl::CProduct::GetNumberOfRecords (const CStringList &datasetNames, CIntMap &datasetRecordsNumber)`
- `virtual double brathl::CProduct::GetOffset ()`
- `CCriteriaPass * brathl::CProduct::GetPassCriteria ()`
- `CCriteriaPassInfo * brathl::CProduct::GetPassCriteriaInfo ()`
- `CStringArray * brathl::CCriteriaPassString::GetPasses ()`
- `CCriteriaPassInt * brathl::CProduct::GetPassIntCriteria ()`
- `CCriteriaPassIntInfo * brathl::CProduct::GetPassIntCriteriaInfo ()`
- `CCriteriaPassString * brathl::CProduct::GetPassStringCriteria ()`
- `CCriteriaPassStringInfo * brathl::CProduct::GetPassStringCriteriaInfo ()`
- `int32_t brathl::CProduct::GetPerformBoundaryChecks ()`
- `int32_t brathl::CProduct::GetPerformConversions ()`
- `string brathl::CProduct::GetProductClass ()`
- `string brathl::CProduct::GetProductClassType ()`
- `void brathl::CMapProduct::GetProductKeysWithCriteria (CStringArray &keys)`
- `CProductList & brathl::CProduct::GetProductList ()`
- `string brathl::CProduct::GetProductType ()`
- `string brathl::CProduct::GetRecordFieldName ()`
- `virtual void brathl::CProduct::GetRecords (CStringArray &array)`
- `brathl_refDate brathl::CProduct::GetRefDate ()`
- `CDate brathl::CProduct::GetRefDateAsDate ()`
- `void brathl::CProduct::GetRootType ()`
- `uint32_t brathl::CProduct::GetSkippedRecordCount ()`
- `int32_t brathl::CCriteriaPassInt::GetTo ()`
- `CTreeField * brathl::CProduct::GetTreeField ()`
- `string brathl::CProduct::GetTypeDesc ()`
- `string brathl::CProduct::GetTypeDesc (coda_Type *type)`
- `string brathl::CProduct::GetTypeName ()`
- `string brathl::CProduct::GetTypeUnit ()`
- `virtual bool brathl::CProduct::GetValueMinMax (CExpression &expr, const string &recordName, double &valueMin, double &valueMax, const CUnit &unit)`
- `static void brathl::CProduct::GroupAliases (const CProduct *product, const CStringMap *formulaAliases, CStringMap &allAliases)`
- `void brathl::CProduct::HandleBratError (const string &str="", int32_t errClass=BRATHL_LOGIC_ERROR)`
- `virtual bool brathl::CProduct::HasAliases ()`
- `virtual bool brathl::CProduct::HasCompatibleDims (const string &value, string &msg, bool useVirtualDims, CUIntArray *commonDimensions=NULL)`

- virtual bool **brathl::CProduct::HasCompatibleDims** (const string &value, const string &dataSetName, string &msg, bool useVirtualDims, **CUIntArray** *commonDimensions=NULL)
- virtual bool **brathl::CProduct::HasCompatibleDims** (const CExpression &expr, string &msg, bool useVirtualDims, **CUIntArray** *commonDimensions=NULL)
- virtual bool **brathl::CProduct::HasCompatibleDims** (const CExpression &expr, const string &dataSetName, string &msg, bool useVirtualDims, **CUIntArray** *commonDimensions=NULL)
- virtual bool **brathl::CProduct::HasCompatibleDims** (const CStringArray *fieldNames, string &msg, bool useVirtualDims, **CUIntArray** *commonDimensions=NULL)
- virtual bool **brathl::CProduct::HasCompatibleDims** (const CStringArray *fieldNames, const string &dataSetName, string &msg, bool useVirtualDims, **CUIntArray** *commonDimensions=NULL)
- virtual bool **brathl::CProduct::HasCriterialInfo** ()
- bool **brathl::CProduct::HasCycleCriteria** ()
- bool **brathl::CProduct::HasCycleCriterialInfo** ()
- bool **brathl::CProduct::HasDatetimeCriteria** ()
- bool **brathl::CProduct::HasDatetimeCriterialInfo** ()
- bool **brathl::CProduct::HasEqualDims** (const string &value, string &msg)
- bool **brathl::CProduct::HasEqualDims** (const string &value, const string &dataSetName, string &msg)
- bool **brathl::CProduct::HasEqualDims** (const CExpression &expr, string &msg)
- bool **brathl::CProduct::HasEqualDims** (const CExpression &expr, const string &dataSetName, string &msg)
- bool **brathl::CProduct::HasEqualDims** (const CStringArray *fieldNames, string &msg)
- bool **brathl::CProduct::HasEqualDims** (const CStringArray *fieldNames, const string &dataSetName, string &msg)
- bool **brathl::CProduct::HasEqualsNumberOfRecord** (const **CIntMap** &datasetRecordsNumber)
- virtual bool **brathl::CProduct::HasHighResolutionFieldCalculation** ()
- bool **brathl::CProduct::HasLatLonCriteria** ()
- bool **brathl::CProduct::HasLatLonCriterialInfo** ()
- bool **brathl::CProduct::HasPassCriteria** ()
- bool **brathl::CProduct::HasPassCriterialInfo** ()
- bool **brathl::CProduct::HasPassIntCriteria** ()
- bool **brathl::CProduct::HasPassIntCriterialInfo** ()
- bool **brathl::CProduct::HasPassStringCriteria** ()
- bool **brathl::CProduct::HasPassStringCriterialInfo** ()
- void **brathl::CCriteriaPass::Init** ()
- void **brathl::CCriteriaPassString::Init** ()
- void **brathl::CCriteriaPassInt::Init** ()
- void **brathl::CMapProduct::Init** ()
- virtual void **brathl::CProduct::InitApplyCriteriaStats** ()
- virtual void **brathl::CProduct::InitCriterialInfo** ()
- virtual void **brathl::CProduct::InitDateRef** ()=0
- virtual void **brathl::CProductGeneric::InitDateRef** ()
- virtual void **brathl::CProduct::InitInternalFieldName** (const string &dataSetName, **CStringList** &listField, bool convertDate=false)
- virtual void **brathl::CProduct::InitInternalFieldName** (**CStringList** &listField, bool convertDate=false)
- virtual void **brathl::CProduct::InitInternalFieldName** (const string &field, bool convertDate=false)
- void **brathl::CProduct::InsertRecord** (int32_t pos)
- void **brathl::CProduct::InsertRecord** (CDataSet &dataSet, int32_t pos)
- bool **brathl::CCriteriaPassString::Intersect** (const string &passes, CStringArray &intersect)
- bool **brathl::CCriteriaPassString::Intersect** (CStringArray &passes, CStringArray &intersect)
- bool **brathl::CCriteriaPassInt::Intersect** (CStringArray &array, CStringArray &intersect)
- bool **brathl::CCriteriaPassInt::Intersect** (CStringArray &array, **CIntArray** &intersect)
- bool **brathl::CCriteriaPassInt::Intersect** (**CIntArray** &array, CStringArray &intersect)
- bool **brathl::CCriteriaPassInt::Intersect** (**CIntArray** &array, **CIntArray** &intersect)
- bool **brathl::CCriteriaPassInt::Intersect** (int32_t from, int32_t to, CStringArray &intersect)
- bool **brathl::CCriteriaPassInt::Intersect** (int32_t from, int32_t to, **CIntArray** &intersect)

- bool **brathl::CCriteriaPassInt::Intersect** (double otherFrom, double otherTo, **CIntArray** &intersect)
- bool **brathl::CCriteriaPassInt::Intersect** (const string &from, const string &to, **CIntArray** &intersect)
- bool **brathl::CCriteriaPassInt::Intersect** (const string &from, const string &to, **CStringArray** &intersect)
- bool **brathl::CProductList::IsATP** ()
- virtual bool **brathl::CCriteriaPass::IsDefaultValue** ()=0
- bool **brathl::CCriteriaPassString::IsDefaultValue** ()
- bool **brathl::CCriteriaPassInt::IsDefaultValue** ()
- bool **brathl::CProductList::IsGenericNetCdf** ()
- bool **brathl::CProductList::IsHdfOrNetcdfCodaFormat** ()
- static bool **brathl::CProductList::IsHdfOrNetcdfCodaFormat** (coda_format format)
- virtual bool **brathl::CProduct::IsHighResolutionField** (**CField** *field)
- bool **brathl::CProductList::IsJason2** ()
- virtual bool **brathl::CProduct::IsLatitudeFieldName** (const string &name)
- virtual bool **brathl::CProduct::IsLongitudeFieldName** (const string &name)
- bool **brathl::CProduct::IsNetCdf** ()
- bool **brathl::CProductList::IsNetCdfCProduct** ()
- bool **brathl::CProduct::IsNetCdfCProduct** ()
- bool **brathl::CProductList::IsNetCdfOrNetCdfCProduct** ()
- bool **brathl::CProduct::IsNetCdfOrNetCdfCProduct** ()
- bool **brathl::CProductList::IsNetCdfProduct** ()
- bool **brathl::CProduct::IsNetCdfProduct** ()
- virtual bool **brathl::CProduct::IsOpened** ()
- virtual bool **brathl::CProduct::IsOpened** (const string &fileName)
- bool **brathl::CProductList::IsSameProduct** (const string &productClass, const string &productType)
- bool **brathl::CProduct::IsSameProduct** (const **CProductList** fileList)
- bool **brathl::CProduct::IsSameProduct** (const string &productClass, const string &productType)
- bool **brathl::CProduct::IsSetCycleCriteria** ()
- bool **brathl::CProduct::IsSetDatetimeCriteria** ()
- bool **brathl::CProduct::IsSetLatLonCriteria** ()
- bool **brathl::CProduct::IsSetPassCriteria** ()
- bool **brathl::CProduct::IsSetPassIntCriteria** ()
- bool **brathl::CProduct::IsSetPassStringCriteria** ()
- bool **brathl::CProductList::IsYFX** ()
- bool **brathl::CProductList::IsZFX** ()
- virtual void **brathl::CProduct::LoadAliases** ()
- virtual void **brathl::CProduct::LoadFieldsInfo** ()
- bool **brathl::CProduct::LoadTransposeFieldsValue** (**CStringArray** &fieldsToTranspose)
- void **brathl::CProduct::Log** (const char *str, bool bCrLf=true)
- void **brathl::CProduct::Log** (const string &str, bool bCrLf=true)
- void **brathl::CProduct::Log** (double n, bool bCrLf=true)
- void **brathl::CProduct::Log** (int32_t n, bool bCrLf=true)
- void **brathl::CProduct::Log** (bool n, bool bCrLf=true)
- void **brathl::CProduct::Log** (const **CStringList** &l, bool bCrLf=true)
- void **brathl::CProduct::LogSelectionResult** (const string &fileName, bool result)
- virtual string **brathl::CProduct::MakeInternalDataSetName** (const string &dataSetName)
- virtual string **brathl::CProduct::MakeInternalFieldName** (const string &dataSetName, const string &field)
- virtual string **brathl::CProduct::MakeInternalFieldName** (const string &field)
- virtual string **brathl::CProduct::MakeInternalNameByAddingRoot** (const string &name)
- virtual bool **brathl::CProduct::Open** (const string &fileName, const string &dataSetName, **CStringList** &list-FieldToRead, bool convertDate=false)
- virtual bool **brathl::CProduct::Open** (const string &fileName, const string &dataSetName)
- virtual bool **brathl::CProduct::Open** (const string &fileName)
- virtual bool **brathl::CProduct::Open** ()
- const **CProductList** & **brathl::CProductList::operator=** (const **CProductList** &lst)
- const **CCriteriaPassString** & **brathl::CCriteriaPassString::operator=** (**CCriteriaPassString** &c)

- const **CCriteriaPassInt** & **bratl::CCriteriaPassInt::operator=** (**CCriteriaPassInt** &c)
- **CInfo** * **bratl::CProduct::CListInfo::PrevBack** (bool withExcept=true)
- void **bratl::CProduct::ProcessHighResolution** ()
- virtual void **bratl::CProduct::ProcessHighResolutionWithFieldCalculation** ()
- virtual void **bratl::CProduct::ProcessHighResolutionWithoutFieldCalculation** ()
- virtual void **bratl::CProduct::Put** (**CDataSet** *dataSet, **CFieldSetDbf** *fieldSetDbf, uint32_t repeat, uint32_t insertRecordAt=0)
- virtual void **bratl::CProduct::Put** (**CDataSet** *dataSet, **CFieldSetArrayDbf** *fieldSetArrayDbf, uint32_t repeat, uint32_t insertRecordAt=0)
- virtual void **bratl::CProduct::Put** (**CDataSet** *dataSet, **CFieldSetDbf** *fieldSetDbf)
- virtual void **bratl::CProduct::PutFlat** (**CDataSet** *dataSet, **CFieldSetArrayDbf** *fieldSetArrayDbf, uint32_t insertRecordAt=0)
- virtual void **bratl::CProduct::PutFlatHighResolution** (**CDataSet** *dataSet, **CFieldSetArrayDbf** *fieldSetArrayDbf)
- virtual void **bratl::CProduct::ReadBratFieldRecord** (const string &key, int32_t iRecord)
- virtual void **bratl::CProduct::ReadBratFieldRecord** (**CField::CListField::iterator** it)
- virtual void **bratl::CProduct::ReadBratFieldRecord** (**CField::CListField::iterator** it, bool &skipRecord)
- virtual void **bratl::CProduct::ReadBratRecord** (const string &dataSetName, const string &field, int32_t iRecord)
- virtual void **bratl::CProduct::ReadBratRecord** (const string &dataSetName, **CStringList** &listField, int32_t iRecord)
- virtual void **bratl::CProduct::ReadBratRecord** (int32_t iRecord)
- static int32_t **bratl::CProduct::ReadData** (int32_t nbFiles, char **fileNames, const char *recordName, const char *selection, int32_t nbData, char **dataExpressions, char **units, double **results, int32_t sizes[], int32_t *actualSize, int ignoreOutOfRange, int statistics, double defaultValue, **CStringMap** *fieldSpecificUnit=NULL)
- static void **bratl::CProduct::ReadDataForOneMeasure** (**CDataSet** *dataSet, const string &recordName, **CExpression** &Select, vector< **CExpression** > &Expressions, const vector< **CUnit** > &WantedUnits, double **results, int32_t *sizes, int32_t *actualSize, int ignoreOutOfRange, int statistics, **CProduct** *product=NULL)
- void **bratl::CProduct::RemoveCriteria** ()
- void **bratl::CMapProduct::RemoveCriteriaFromProducts** ()
- void **bratl::CProduct::RemoveUnusedFields** ()
- void **bratl::CProduct::ReplaceNamesCaseSensitive** (const **CExpression** &exprIn, const **CStringArray** &fieldsIn, **CExpression** &exprOut, bool forceReload=false)
- void **bratl::CProduct::ReplaceNamesCaseSensitive** (const string &in, const **CStringArray** &fieldsIn, string &out, bool forceReload=false)
- void **bratl::CProduct::ReplaceNamesCaseSensitive** (const string &in, string &out, bool forceReload=false)
- void **bratl::CProduct::ReplaceNamesCaseSensitive** (const **CExpression** &exprIn, string &out, bool forceReload=false)
- virtual void **bratl::CProduct::Rewind** ()
- virtual void **bratl::CProduct::RewindEnd** ()
- virtual void **bratl::CProduct::RewindInit** ()
- virtual void **bratl::CProduct::RewindProcess** ()
- void **bratl::CProductList::Set** (const **CProductList** &lst)
- void **bratl::CCriteriaPassString::Set** (const string &passes, const string &delimiter=CCriteriaPassString::m_delimiter)
- void **bratl::CCriteriaPassString::Set** (const **CStringArray** &array)
- void **bratl::CCriteriaPassString::Set** (**CCriteriaPassString** &c)
- void **bratl::CCriteriaPassInt::Set** (**CCriteriaPassInt** &c)
- void **bratl::CCriteriaPassInt::Set** (int32_t from, int32_t to)
- void **bratl::CCriteriaPassInt::Set** (const string &from, const string &to)
- void **bratl::CCriteriaPassInt::Set** (const **CStringArray** &array)
- void **bratl::CProduct::SetCreateVirtualField** (bool value)
- void **bratl::CProduct::SetCursor** (**CField** *field, bool &skipRecord)
- void **bratl::CProduct::SetDataSetNameToRead** (const string &value)

- virtual void **brathl::CCriteriaPass::SetDefaultValue** ()=0
- void **brathl::CCriteriaPassString::SetDefaultValue** ()
- void **brathl::CCriteriaPassInt::SetDefaultValue** ()
- void **brathl::CProduct::SetDescription** (const string &value)
- void **brathl::CProduct::SetDisableTrace** (bool value)
- void **brathl::CProduct::SetDynInfo** ()
- void **brathl::CProduct::SetExpandArray** (bool value)
- void **brathl::CProduct::SetFieldSpecificUnit** (const string &key, const string &value)
- virtual void **brathl::CProduct::SetFieldSpecificUnit** (CField *field)
- void **brathl::CProduct::SetFieldSpecificUnits** (const CStringMap &fieldSpecificUnit)
- virtual void **brathl::CProduct::SetForceReadDataOneByOne** (bool value)
- void **brathl::CCriteriaPassInt::SetFrom** (int32_t from)
- void **brathl::CCriteriaPassInt::SetFrom** (const string &from)
- void **brathl::CCriteriaPassInt::SetFromText** (const string &values, const string &delimiter=CCriteriaPassInt::m_delimiter)
- virtual void **brathl::CProduct::SetHighResolution** (CField *field)
- void **brathl::CProduct::SetIndex** (CField *field)
- virtual void **brathl::CProduct::SetLabel** (const string &value)
- void **brathl::CProduct::SetListFieldOrigin** (const CStringList &listFieldOrigin)
- void **brathl::CProduct::SetListFieldToRead** (CStringList &listFieldToRead, bool convertDate=false)
- void **brathl::CProduct::SetNativeType** (CField *field)
- virtual void **brathl::CProduct::SetOffset** (double value)
- void **brathl::CProduct::SetPerformBoundaryChecks** (bool performBoundaryChecks)
- void **brathl::CProduct::SetPerformConversions** (bool performConversions)
- void **brathl::CProduct::SetProductList** (const string &fileName, bool checkFiles=true)
- void **brathl::CProduct::SetProductList** (const CStringList &fileList, bool checkFiles=true)
- void **brathl::CProduct::SetSpecialType** (CField *field)
- void **brathl::CCriteriaPassInt::SetTo** (int32_t to)
- void **brathl::CCriteriaPassInt::SetTo** (const string &to)
- void **brathl::CProduct::SetTypeClass** (CField *field)
- void **brathl::CProduct::SetUnion** (CField *field)
- bool **brathl::CProduct::TraverseData** ()
- bool **brathl::CProduct::TraverseRecord** (int32_t indexFields)
- virtual **brathl::CCriteriaPass::~~CCriteriaPass** ()
Destructor.
- virtual **brathl::CCriteriaPassInt::~~CCriteriaPassInt** ()
Destructor.
- virtual **brathl::CCriteriaPassString::~~CCriteriaPassString** ()
Destructor.
- virtual **brathl::CMapProduct::~~CMapProduct** ()
CIntMap (p. 257) *dtor.*
- virtual **brathl::CProduct::~~CProduct** ()
Destructor.
- virtual **brathl::CProductGeneric::~~CProductGeneric** ()
Destructor.
- virtual **brathl::CProductList::~~CProductList** ()
Destructor.

Variables

- static const uint32_t **brathl::CProduct::COUNT_INDEX** = 0
- const long **brathl::DEFAULT_DIM** [] = {1}
- CStringArray **brathl::CProduct::m_arrayLatitudeFieldName**
- CStringArray **brathl::CProduct::m_arrayLongitudeFieldName**
- static coda_array_ordering **brathl::CProduct::m_arrayOrdering** = coda_array_ordering_c
- uint32_t **brathl::CProduct::m_countForTrace**
- bool **brathl::CProduct::m_createVirtualField**
- COblntMap **brathl::CProduct::m_criterialInfoMap**
- COblntMap **brathl::CProduct::m_criteriaMap**
- int32_t **brathl::CProduct::m_currentRecord**
- coda_ProductFile * **brathl::CProduct::m_currFile**
- string **brathl::CProduct::m_currFileName**
- coda_Cursor **brathl::CProduct::m_cursor**
- CStringArray **brathl::CProduct::m_dataDictionaryFieldNames**
- CStringArray **brathl::CProduct::m_dataDictionaryFieldNamesWithDatasetName**
- CDataSet **brathl::CProduct::m_dataSet**
- string **brathl::CProduct::m_dataSetNameToRead**
- CDate **brathl::CProduct::m_dateProcessBegin**
- CDate **brathl::CProduct::m_dateProcessEnd**
- static const string **brathl::CCriteriaPassString::m_delimiter** = ","
- static const string **brathl::CCriteriaPassInt::m_delimiter** = " "
- double **brathl::CProduct::m_deltaTimeHighResolution**
- string **brathl::CProduct::m_description**
- bool **brathl::CProduct::m_disableTrace**
- bool **brathl::CProduct::m_expandArray**
- string **brathl::CProduct::CInfo::m_fieldName**
- CStringMap **brathl::CProduct::m_fieldNameEquivalence**
- bool **brathl::CProduct::m_fieldsHaveDefaultValue**
- CStringMap **brathl::CProduct::m_fieldSpecificUnit**
- CStringList **brathl::CProduct::m_fieldsToProcess**
- CStringArray **brathl::CProduct::m_fieldsToTranspose**
- CProductList **brathl::CProduct::m_fileList**
- double **brathl::CProduct::m_forceLatMaxCriteriaValue**
- double **brathl::CProduct::m_forceLatMinCriteriaValue**
- int32_t **brathl::CCriteriaPassInt::m_from**
- bool **brathl::CProduct::m_hasHighResolutionFieldToProcess**
- int32_t **brathl::CProduct::CInfo::m_index**
- int32_t **brathl::CProduct::m_indexProcessedFile**
- int32_t **brathl::CProduct::CInfo::m_isUnion**
- string **brathl::CProduct::m_label**
- string **brathl::CProduct::m_latitudeFieldName**
- CStringList **brathl::CProduct::m_listFieldExpandArray**
- CStringList **brathl::CProduct::m_listFieldOrigin**
- CField::CListField **brathl::CProduct::m_listFields**
- CListInfo **brathl::CProduct::m_listInfo**
- CStringList **brathl::CProduct::m_listInternalFieldName**
- CFile * **brathl::CProduct::m_logFile**
- string **brathl::CProduct::m_longitudeFieldName**
- CStringMap **brathl::CProduct::m_mapStringAliases**
- string **brathl::CProductList::m_message**
- int32_t **brathl::CProduct::m_nbRecords**
- uint32_t **brathl::CProduct::m_nSkippedRecord**
- uint32_t **brathl::CProduct::m_numHighResolutionMeasure**

- double **brathl::CProduct::m_offset**
- CStringArray **brathl::CCriteriaPassString::m_passes**
- double **brathl::CProduct::m_previousLatitude**
- double **brathl::CProduct::m_previousLongitude**
- double **brathl::CProduct::m_previousTimeStamp**
- CProductAliases * **brathl::CProduct::m_productAliases**
- string **brathl::CProductList::m_productClass**
- coda_format **brathl::CProductList::m_productFormat**
- string **brathl::CProductList::m_productType**
- int32_t **brathl::CProduct::m_recordCount**
- **brathl_refDate** **brathl::CProduct::m_refDate**
- int32_t **brathl::CProduct::m_refPoint**
- int32_t **brathl::CCriteriaPassInt::m_to**
- uint32_t **brathl::CProduct::m_traceProcessRecordRatio**
- static const char * **brathl::CProduct::m_transposeFieldValuesFileName** = "brathl_transposefieldvalues.txt"
- **CTreeField** **brathl::CProduct::m_tree**
- static const string **brathl::CProduct::m_treeRootName** = "Root"
- coda_Type * **brathl::CProduct::CInfo::m_type**
- coda_type_class **brathl::CProduct::CInfo::m_type_class**
- static const uint32_t **brathl::CProduct::MAX_INDEX** = 4
- static const uint32_t **brathl::CProduct::MEAN_INDEX** = 1
- static const uint32_t **brathl::CProduct::MIN_INDEX** = 3
- static const int32_t **brathl::CProduct::NUMBER_OF_STATISTICS** = 5
- static const uint32_t **brathl::CProduct::STDDEV_INDEX** = 2

5.6.1 Detailed Description

5.6.2 Function Documentation

5.6.2.1 **brathl::CCriteriaPassInt::CCriteriaPassInt** (int32_t *from*, int32_t *to*)

Constructor.

Parameters

<i>from</i>	start pass
<i>to</i>	end pass

5.6.2.2 **brathl::CCriteriaPassInt::CCriteriaPassInt** (const string & *from*, const string & *to*)

Constructor.

Parameters

<i>from</i>	start pass
<i>to</i>	end pass

5.6.2.3 **brathl::CCriteriaPassInt::CCriteriaPassInt** (const CStringArray & *array*)

Constructor from a array that contains start pass as string, end pass as string

Parameters

<i>array</i>	start and end dates
--------------	---------------------

5.6.2.4 `brathl::CCriteriaPassString::CCriteriaPassString (const string & passes, const string & delimiter = CCriteriaPassString::m_delimiter)`

Constructor from a string that contains passes delimited by a comma)

Parameters

<i>passes</i>	passes to set
---------------	---------------

5.6.2.5 `brathl::CCriteriaPassString::CCriteriaPassString (const CStringArray & array)`

Constructor from a array that contains passes

Parameters

<i>array</i>	start and end dates
--------------	---------------------

5.6.2.6 `brathl::CProduct::CProduct (const string & fileName)`

Creates new CProduct object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

5.6.2.7 `brathl::CProduct::CProduct (const CStringList & fileNameList)`

Creates new CProduct object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

5.6.2.8 `brathl::CProductGeneric::CProductGeneric (const string & fileName) [inline]`

Creates new CProdCProductGenericuct object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

5.6.2.9 `brathl::CProductGeneric::CProductGeneric (const CStringList & fileNameList) [inline]`

Creates new CProductGeneric object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

5.6.2.10 `brathl::CProductList::CProductList (const CProductList & p)`

Creates new **CProductList** (p. 298) object from another one

Parameters

<i>p</i>	[in] : productList object to be connected
----------	---

5.6.2.11 `brathl::CProductList::CProductList (const string & fileName)`

Creates new **CProductList** (p. 298) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

5.6.2.12 `brathl::CProductList::CProductList (const CStringList & fileNameList)`

Creates new CProduct object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

5.6.2.13 `brathl::CProductList::CProductList (const CStringArray & fileNameArray)`

Creates new CProduct object

Parameters

<i>fileNameArray</i>	[in] : array of file to be connected
----------------------	--------------------------------------

5.6.2.14 `bool brathl::CCriteriaPassString::Intersect (const string & passes, CStringArray & intersect)`

Creates the intersection of these passes with the given one

Parameters

<i>passes</i>	intersect with this
<i>intersect</i>	intersection passes

Returns

true, or false if there is no intersection

5.6.2.15 `bool brathl::CCriteriaPassString::Intersect (CStringArray & passes, CStringArray & intersect)`

Creates the intersection of these passes with the given one

Parameters

<i>passes</i>	intersect with this
<i>intersect</i>	intersection passes

Returns

true, or false if there is no intersection

5.6.2.16 `bool brathl::CCriteriaPassInt::Intersect (CStringArray & array, CStringArray & intersect)`

Create the intersection of this date period with the given one

Parameters

<i>array</i>	that contains start pass as string, end pass as string
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

5.6.2.17 `bool brathl::CCriteriaPassInt::Intersect (CStringArray & array, CIntArray & intersect)`

Create the intersection of this date period with the given one

Parameters

<i>array</i>	that contains start pass as string, end pass as string
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

5.6.2.18 `bool brathl::CCriteriaPassInt::Intersect (CIntArray & array, CStringArray & intersect)`

Create the intersection of this date period with the given one

Parameters

<i>array</i>	that contains start pass as string, end pass as string
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

5.6.2.19 `bool brathl::CCriteriaPassInt::Intersect (CIntArray & array, CIntArray & intersect)`

Create the intersection of this date period with the given one

Parameters

<i>array</i>	that contains start pass as string, end pass as string
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

5.6.2.20 `virtual bool brathl::CCriteriaPass::IsDefaultValue () [pure virtual]`

Tests whether date period have been initialized or not

Returns

true if not initialized

Implements **brathl::CCriteria** (p. 138).

Implemented in **brathl::CCriteriaPassInt** (p. 80), and **brathl::CCriteriaPassString** (p. 79).

5.6.2.21 `bool brathl::CCriteriaPassString::IsDefaultValue () [virtual]`

Tests whether passes have been initialized or not

Returns

true if not initialized

Implements **brathl::CCriteriaPass** (p. 79).

5.6.2.22 **bool** brathl::CCriteriaPassInt::IsDefaultValue () [virtual]

Tests whether the pass have been initialized or not

Returns

true if not initialized

Implements **brathl::CCriteriaPass** (p. 79).

5.6.2.23 **virtual bool** brathl::CProduct::IsHighResolutionField (**CField** * *field*) [inline],[virtual]

Determines if a field object is a 'high resolution' array data see classes derived from CProduct.

5.6.2.24 **void** brathl::CProductList::Set (**const CProductList** & *lst*)

Creates new **CProductList** (p. 298) object from another one

Parameters

<i>p</i>	[in] : productList object to be connected
----------	---

References brathl::CStringList::operator=().

5.6.2.25 **void** brathl::CCriteriaPassString::Set (**const string** & *passes*, **const string** & *delimiter* = **CCriteriaPassString::m_delimiter**)

Sets one or more passes from a string (delimited by a comma)

Parameters

<i>passes</i>	passes to set
---------------	---------------

5.6.2.26 **void** brathl::CCriteriaPassString::Set (**const CStringArray** & *array*)

Sets passes from a array

Parameters

<i>array</i>	array of passes
--------------	-----------------

5.6.2.27 **void** brathl::CCriteriaPassInt::Set (**int32_t** *from*, **int32_t** *to*)

Sets date period from start and end pass

Parameters

<i>from</i>	start pass
<i>to</i>	end pass

5.6.2.28 **void** brathl::CCriteriaPassInt::Set (**const string** & *from*, **const string** & *to*)

Sets date period from start and end pass

Parameters

<i>from</i>	start pass
<i>to</i>	end pass

References `brathl::CTools::StrToInt()`.

5.6.2.29 `void brathl::CCriteriaPassInt::Set (const CStringArray & array)`

Sets a date period from a array that contains start pass as string, end pass as string

Parameters

<i>array</i>	start and end dates
--------------	---------------------

5.6.2.30 `virtual void brathl::CCriteriaPass::SetDefaultValue () [pure virtual]`

Sets internal value to the default value (uninitialized)

Implements **`brathl::CCriteria`** (p. 139).

Implemented in **`brathl::CCriteriaPassInt`** (p. 81), and **`brathl::CCriteriaPassString`** (p. 81).

5.6.2.31 `void brathl::CCriteriaPassString::SetDefaultValue () [virtual]`

Sets internal value to the default value (uninitialized)

Implements **`brathl::CCriteriaPass`** (p. 81).

5.6.2.32 `void brathl::CCriteriaPassInt::SetDefaultValue () [virtual]`

Sets internal value to the default value (uninitialized)

Implements **`brathl::CCriteriaPass`** (p. 81).

5.6.2.33 `void brathl::CCriteriaPassInt::SetFrom (int32_t from)`

Sets start pass

Parameters

<i>to</i>	start pass
-----------	------------

5.6.2.34 `void brathl::CCriteriaPassInt::SetFrom (const string & from)`

Sets start pass

Parameters

<i>to</i>	start pass
-----------	------------

References `brathl::CTools::StrToInt()`.

5.6.2.35 `void brathl::CCriteriaPassInt::SetTo (int32_t to)`

Sets end pass

Parameters

<i>to</i>	end pass
-----------	----------

5.6.2.36 void brathl::CCriteriaPassInt::SetTo (const string & to)

Sets end pass

Parameters

to	end pass
----	----------

References brathl::CTools::StrToInt().

5.6.3 Variable Documentation

5.6.3.1 const long brathl::DEFAULT_DIM[] = {1}

Product management class.

Version

1.0

5.6.3.2 int32_t brathl::CCriteriaPassInt::m_from [protected]

start pass

5.6.3.3 int32_t brathl::CProduct::m_nbRecords [protected]

Number of records to read

5.6.3.4 CStringArray brathl::CCriteriaPassString::m_passes [protected]

Date period

5.6.3.5 int32_t brathl::CCriteriaPassInt::m_to [protected]

end pass

5.7 Date conversion classes

Classes

- class **brathl::CDate**
- class **brathl::CDatePeriod**
- class **brathl::CMission**

5.7.1 Detailed Description

5.8 Errors management

Classes

- class **brathl::CAlgorithmException**
- class **brathl::CException**
- class **brathl::CExpressionException**
- class **brathl::CFileException**
- class **brathl::CLoadAliasesException**
- class **brathl::CMemoryException**
- class **brathl::CParameterException**
- class **brathl::CProductException**
- class **brathl::CUnImplementException**
- class **brathl::CXMLException**
- class **brathl::CXMLParseException**

5.8.1 Detailed Description

5.9 File services

Classes

- class **brathl::CFile**

5.9.1 Detailed Description

5.10 Parameters

Classes

- class **brathl::CFileParams**
- class **brathl::CMapParameter**
- class **brathl::CParameter**

Functions

- **brathl::CMapParameter::CMapParameter ()**
CMapParameter (p. 262) ctor.
- virtual void **brathl::CMapParameter::Dump** (ostream &fOut=cerr)
Dump function.
- bool **brathl::CMapParameter::Erase** (CMapParameter::iterator iteratorParameter)
- bool **brathl::CMapParameter::Erase** (const string &key)
- **CParameter * brathl::CMapParameter::Exists** (const string &key)
- **CParameter * brathl::CMapParameter::Insert** (const string &key, const string &value)
- **CParameter * brathl::CMapParameter::operator[]** (const string key)
- void **brathl::CMapParameter::RemoveAll** ()
- virtual **brathl::CMapParameter::~~CMapParameter** ()
CMapParameter (p. 262) dtor.

5.10.1 Detailed Description

5.10.2 Function Documentation

5.10.2.1 bool brathl::CMapParameter::Erase (CMapParameter::iterator iteratorParameter)

Delete an element referenced by iteratorMnemo

Returns

true if no error, otherwise false

5.10.2.2 bool brathl::CMapParameter::Erase (const string & key)

Delete an element by its key

Returns

true if no error, otherwise false

5.10.2.3 CParameter * brathl::CMapParameter::Exists (const string & key)

Tests if an element identify by 'key' already exists

Returns

a **CParameter** (p. 275) pointer if exists, otherwise NULL

5.10.2.4 CParameter * brathl::CMapParameter::Insert (const string & key, const string & value)

Inserts a **CParameter** (p. 275) object

Parameters

<i>key</i>	: parameter name (map key)
<i>value</i>	: parameter value

Returns

CParameter (p. 275) object or NULL if error

References `bratl::CParameter::AddValue()`.

5.10.2.5 CParameter * bratl::CMapParameter::operator[] (const string key)

`operator[]` redefinition. Searches a **CParameter** (p. 275) object identify by 'key'. DON'T USE this syntax if you are not sure the key exists, there's a bug in STL, after calling '`record = m_recordSetMap[recordSetName]`', if key not existed and the map is empty then the key exists in the map and points to a NULL object **CParameter** (p. 275) `*p = m_mapParam[key] -> use Exists method instead ;`

Parameters

<i>key</i>	: parameter keyword
------------	---------------------

Returns

a pointer to th **CParameter** (p. 275) object if found, NULL if not found

5.10.2.6 void bratl::CMapParameter::RemoveAll ()

Remove all elements and clear the map

Referenced by `bratl::CFileParams::Load()`.

5.11 Date conversion C APIs

Functions

- LIBRATHL_API int32_t **brathl_Cycle2YMDHMSM** (**brathl_mission** mission, uint32_t cycle, uint32_t pass, **brathl_DateYMDHMSM** *dateYMDHMSM)
- LIBRATHL_API int32_t **brathl_DayOfYear** (**brathl_DateYMDHMSM** *dateYMDHMSM, uint32_t *dayOfYear)
- LIBRATHL_API int32_t **brathl_DiffDSM** (**brathl_DateDSM** *dateDSM1, **brathl_DateDSM** *dateDSM2, double *diff)
- LIBRATHL_API int32_t **brathl_DiffJulian** (**brathl_DateJulian** *dateJulian1, **brathl_DateJulian** *dateJulian2, double *diff)
- LIBRATHL_API int32_t **brathl_DiffYMDHMSM** (**brathl_DateYMDHMSM** *dateYMDHMSM1, **brathl_DateYMDHMSM** *dateYMDHMSM2, double *diff)
- LIBRATHL_API int32_t **brathl_DSM2Julian** (**brathl_DateDSM** *dateDSM, **brathl_refDate** refDate, **brathl_DateJulian** *dateJulian)
- LIBRATHL_API int32_t **brathl_DSM2Seconds** (**brathl_DateDSM** *dateDSM, **brathl_refDate** refDate, **brathl_DateSecond** *dateSeconds)
- LIBRATHL_API int32_t **brathl_DSM2YMDHMSM** (**brathl_DateDSM** *dateDSM, **brathl_DateYMDHMSM** *dateYMDHMSM)
- LIBRATHL_API int32_t **brathl_Julian2DSM** (**brathl_DateJulian** *dateJulian, **brathl_refDate** refDate, **brathl_DateDSM** *dateDSM)
- LIBRATHL_API int32_t **brathl_Julian2Seconds** (**brathl_DateJulian** *dateJulian, **brathl_refDate** refDate, **brathl_DateSecond** *dateSeconds)
- LIBRATHL_API int32_t **brathl_Julian2YMDHMSM** (**brathl_DateJulian** *dateJulian, **brathl_DateYMDHMSM** *dateYMDHMSM)
- LIBRATHL_API int32_t **brathl_NowYMDHMSM** (**brathl_DateYMDHMSM** *dateYMDHMSM)
- LIBRATHL_API int32_t **brathl_Seconds2DSM** (**brathl_DateSecond** *dateSeconds, **brathl_refDate** refDate, **brathl_DateDSM** *dateDSM)
- LIBRATHL_API int32_t **brathl_Seconds2Julian** (**brathl_DateSecond** *dateSeconds, **brathl_refDate** refDate, **brathl_DateJulian** *dateJulian)
- LIBRATHL_API int32_t **brathl_Seconds2YMDHMSM** (**brathl_DateSecond** *dateSeconds, **brathl_DateYMDHMSM** *dateYMDHMSM)
- LIBRATHL_API int32_t **brathl_YMDHMSM2Cycle** (**brathl_mission** mission, **brathl_DateYMDHMSM** *dateYMDHMSM, uint32_t *cycle, uint32_t *pass)
- LIBRATHL_API int32_t **brathl_YMDHMSM2DSM** (**brathl_DateYMDHMSM** *dateYMDHMSM, **brathl_refDate** refDate, **brathl_DateDSM** *dateDSM)
- LIBRATHL_API int32_t **brathl_YMDHMSM2Julian** (**brathl_DateYMDHMSM** *dateYMDHMSM, **brathl_refDate** refDate, **brathl_DateJulian** *dateJulian)
- LIBRATHL_API int32_t **brathl_YMDHMSM2Seconds** (**brathl_DateYMDHMSM** *dateYMDHMSM, **brathl_refDate** refDate, **brathl_DateSecond** *dateSeconds)

5.11.1 Detailed Description

5.11.2 Function Documentation

5.11.2.1 LIBRATHL_API int32_t brathl_Cycle2YMDHMSM (brathl_mission mission, uint32_t cycle, uint32_t pass, brathl_DateYMDHMSM * dateYMDHMSM)

Converts a cyle/pass into a date

Parameters

in	<i>mission</i>	: mission type (see brathl_mission (p. 372))
in	<i>cycle</i>	: number of cycle to convert
in	<i>pass</i>	: number of pass in the cycle to convert
out	<i>dateYMDHMSM</i>	: date corresponding to the cycle/pass

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Cycle/date conversion error codes** (p. 19))

References **BRATHL_SUCCESS**, `brathl::CMission::Convert()`, `brathl::CDate::Convert2YMDHMSM()`, and `brathl::CMission::CtrlMission()`.

5.11.2.2 **LIBRATHL_API** `int32_t brathl_DayOfYear (brathl_DateYMDHMSM * dateYMDHMSM, uint32_t * dayOfYear)`

Retrieves the day of year of a date

Parameters

in	<i>dateYMDHMSM</i>	: date
out	<i>dayOfYear</i>	: day of year of the date parameter

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::DayOfYear()`, and `brathl::CDate::SetDate()`.

5.11.2.3 **LIBRATHL_API** `int32_t brathl_DiffDSM (brathl_DateDSM * dateDSM1, brathl_DateDSM * dateDSM2, double * diff)`

Computes the difference between two dates (date1 - date2) the result is expressed in a decimal number of seconds

Parameters

in	<i>dateDSM1</i>	: date1
in	<i>dateDSM2</i>	: date2
out	<i>diff</i>	: difference in seconds (date1 - date2)

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, and `brathl::CDate::SetDate()`.

5.11.2.4 **LIBRATHL_API** `int32_t brathl_DiffJulian (brathl_DateJulian * dateJulian1, brathl_DateJulian * dateJulian2, double * diff)`

Computes the difference between two dates (date1 - date2) the result is expressed in a decimal number of seconds

Parameters

in	<i>dateJulian1</i>	: date1
in	<i>dateJulian2</i>	: date2
out	<i>diff</i>	: difference in seconds (date1 - date2)

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, and `brathl::CDate::SetDate()`.

5.11.2.5 **LIBRATHL_API** `int32_t brathl_DiffYMDHMSM (brathl_DateYMDHMSM * dateYMDHMSM1, brathl_DateYMDHMSM * dateYMDHMSM2, double * diff)`

Computes the difference between two dates (date1 - date2) the result is expressed in a decimal number of seconds

Parameters

in	<i>dateYMDHMS-M1</i>	: date1
in	<i>dateYMDHMS-M2</i>	: date2
out	<i>diff</i>	: difference in seconds (date1 - date2)

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS, and brathl::CDate::SetDate().

5.11.2.6 `LIBRATHL_API int32_t brathl_DSM2Julian (brathl_DateDSM * dateDSM, brathl_refDate refDate, brathl_DateJulian * dateJulian)`

Converts a days-seconds-microseconds date into a decimal julian date, according to refDate parameter

Parameters

in	<i>dateDSM</i>	: date to convert
in	<i>refDate</i>	: date reference conversion
out	<i>dateJulian</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS, brathl::CDate::Convert2DecimalJulian(), `_structDateJulian::julian`, `_structDateJulian::refDate`, and brathl::CDate::SetDate().

5.11.2.7 `LIBRATHL_API int32_t brathl_DSM2Seconds (brathl_DateDSM * dateDSM, brathl_refDate refDate, brathl_DateSecond * dateSeconds)`

Converts a date in days-seconds-microseconds into a seconds, according to refDate parameter

Parameters

in	<i>dateDSM</i>	: date to convert
in	<i>refDate</i>	: date reference conversion
out	<i>dateSeconds</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS, brathl::CDate::Convert2Second(), `_structDateSecond::nbSeconds`, `_structDateSecond::refDate`, and brathl::CDate::SetDate().

5.11.2.8 `LIBRATHL_API int32_t brathl_DSM2YMDHMSM (brathl_DateDSM * dateDSM, brathl_DateYMDHMSM * dateYMDHMSM)`

Converts a days-seconds-microseconds date into a year, month, day, hour, minute, second, microsecond date

Parameters

in	<i>dateDSM</i>	: date to convert
out	<i>dateYMDHMSM</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2YMDHMSM()`, and `brathl::CDate::SetDate()`.

5.11.2.9 **LIBRATHL_API** `int32_t brathl_Julian2DSM (brathl_DateJulian * dateJulian, brathl_refDate refDate, brathl_DateDSM * dateDSM)`

Converts a decimal julian date into a days-seconds-microseconds date, according to *refDate* parameter

Parameters

in	<i>dateJulian</i>	: date to convert
in	<i>refDate</i>	: date reference conversion
out	<i>dateDSM</i>	: result of conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2DSM()`, `_structDateDSM::days`, `_structDateDSM::muSeconds`, `_structDateDSM::refDate`, `_structDateDSM::seconds`, and `brathl::CDate::SetDate()`.

5.11.2.10 **LIBRATHL_API** `int32_t brathl_Julian2Seconds (brathl_DateJulian * dateJulian, brathl_refDate refDate, brathl_DateSecond * dateSeconds)`

Converts a decimal julian date into seconds, according to *refDate* parameter

Parameters

in	<i>dateJulian</i>	: date to convert
in	<i>refDate</i>	: date reference conversion
out	<i>dateSeconds</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2Second()`, `_structDateSecond::nbSeconds`, `_structDateSecond::refDate`, and `brathl::CDate::SetDate()`.

5.11.2.11 **LIBRATHL_API** `int32_t brathl_Julian2YMDHMSM (brathl_DateJulian * dateJulian, brathl_DateYMDHMSM * dateYMDHMSM)`

Converts a decimal julian date into a year, month, day, hour, minute, second, microsecond date

Parameters

in	<i>dateJulian</i>	: date to convert
out	<i>dateYMDHMSM</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2YMDHMSM()`, and `brathl::CDate::SetDate()`.

5.11.2.12 `LIBRATHL_API int32_t brathl_NowYMDHMSM (brathl_DateYMDHMSM * dateYMDHMSM)`

Gets the current date/time,

Parameters

out	<i>dateYMDHMSM</i>	: current date/time
-----	--------------------	---------------------

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2YMDHMSM()`, and `brathl::CDate::SetDateNow()`.

5.11.2.13 `LIBRATHL_API int32_t brathl_Seconds2DSM (brathl_DateSecond * dateSeconds, brathl_refDate refDate, brathl_DateDSM * dateDSM)`

Converts seconds into a days-seconds-microseconds date, according to `refDate` parameter

Parameters

in	<i>dateSeconds</i>	: date to convert
in	<i>refDate</i>	: date reference conversion
out	<i>dateDSM</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2DSM()`, `_structDateDSM::days`, `_structDateDSM::muSeconds`, `_structDateDSM::refDate`, `_structDateDSM::seconds`, and `brathl::CDate::SetDate()`.

5.11.2.14 `LIBRATHL_API int32_t brathl_Seconds2Julian (brathl_DateSecond * dateSeconds, brathl_refDate refDate, brathl_DateJulian * dateJulian)`

Converts seconds into a decimal julian date, according to `refDate` parameter

Parameters

in	<i>dateSeconds</i>	: date to convert
in	<i>refDate</i>	: date reference conversion
out	<i>dateJulian</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2DecimalJulian()`, `_structDateJulian::julian`, `_structDateJulian::refDate`, and `brathl::CDate::SetDate()`.

5.11.2.15 `LIBRATHL_API int32_t brathl_Seconds2YMDHMSM (brathl_DateSecond * dateSeconds, brathl_DateYMDHMSM * dateYMDHMSM)`

Converts seconds into a year, month, day, hour, minute, second, microsecond date

Parameters

in	<i>dateSeconds</i>	: date to convert
out	<i>dateYMDHMSM</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2YMDHMSM()`, and `brathl::CDate::SetDate()`.

5.11.2.16 `LIBRATHL_API int32_t brathl_YMDHMSM2Cycle (brathl_mission mission, brathl_DateYMDHMSM * dateYMDHMSM, uint32_t * cycle, uint32_t * pass)`

Converts a date into a cycle/pass

Parameters

in	<i>mission</i>	: mission type (see brathl_mission (p. 372))
in	<i>dateYMDHMSM</i>	: date to convert
out	<i>cycle</i>	: number of cycle
out	<i>pass</i>	: number of pass in the cycle

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Cycle/date conversion error codes** (p. 19))

References **BRATHL_SUCCESS**, `brathl::CMission::Convert()`, `brathl::CMission::CtrlMission()`, and `brathl::CDate::SetDate()`.

5.11.2.17 `LIBRATHL_API int32_t brathl_YMDHMSM2DSM (brathl_DateYMDHMSM * dateYMDHMSM, brathl_refDate refDate, brathl_DateDSM * dateDSM)`

Converts a year, month, day, hour, minute, second, microsecond date into a days-seconds-microseconds date, according to refDate parameter

Parameters

in	<i>dateYMDHMSM</i>	: date to convert
in	<i>refDate</i>	: date reference conversion
out	<i>dateDSM</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2DSM()`, `_structDateDSM::days`, `_structDateDSM::muSeconds`, `_structDateDSM::refDate`, `_structDateDSM::seconds`, and `brathl::CDate::SetDate()`.

5.11.2.18 `LIBRATHL_API int32_t brathl_YMDHMSM2Julian (brathl_DateYMDHMSM * dateYMDHMSM, brathl_refDate refDate, brathl_DateJulian * dateJulian)`

Converts a year, month, day, hour, minute, second, microsecond date into a decimal julian date, according to refDate parameter

Parameters

in	<i>dateYMDHMSM</i>	: date to convert
in	<i>refDate</i>	: date reference conversion
out	<i>dateJulian</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2DecimalJulian()`, `_structDateJulian::julian`, `_structDateJulian::refDate`, and `brathl::CDate::SetDate()`.

5.11.2.19 **LIBRATHL_API** `int32_t brathl_YMDHMSM2Seconds (brathl_DateYMDHMSM * dateYMDHMSM, brathl_refDate refDate, brathl_DateSecond * dateSeconds)`

Converts a year, month, day, hour, minute, second, microsecond date into seconds, according to `refDate` parameter

Parameters

in	<i>dateYMDHMSM</i>	: date to convert
in	<i>refDate</i>	: date reference conversion
out	<i>dateSeconds</i>	: result of the conversion

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, `brathl::CDate::Convert2Second()`, `_structDateSecond::nbSeconds`, `_structDateSecond::refDate`, and `brathl::CDate::SetDate()`.

5.12 C API for reading data

Functions

- LIBRATHL_API void **brathl_LoadAliasesDictionary** ()
- LIBRATHL_API int32_t **brathl_ReadData** (int32_t nbFiles, char **fileNames, const char *recordName, const char *selection, int32_t nbData, char **dataExpressions, char **units, double **results, int32_t sizes[], int32_t *actualSize, int ignoreOutOfRange, int statistics, double defaultValue)
- LIBRATHL_API void **brathl_RegisterAlgorithms** ()

5.12.1 Detailed Description

5.12.2 Function Documentation

5.12.2.1 LIBRATHL_API int32_t **brathl_ReadData** (int32_t *nbFiles*, char ** *fileNames*, const char * *recordName*, const char * *selection*, int32_t *nbData*, char ** *dataExpressions*, char ** *units*, double ** *results*, int32_t *sizes*[], int32_t * *actualSize*, int *ignoreOutOfRange*, int *statistics*, double *defaultValue*)

Read data from a set of files Each measure for a data is a scalar value (a single number)

Parameters

in	<i>nbFiles</i>	: Number of files in file name list This is the usable size of #fileNames
in	<i>fileNames</i>	: File name list Must contain at least #nbFiles entries. If an entry is NULL or points to an empty string, the entry is ignored.
in	<i>selection</i>	: Expression involving data fields which has to be true to select returned data if NULL or empty string no selection is done (all data is selected)
in	<i>nbData</i>	: Number of expression used to retrieve data
in	<i>dataExpressions</i>	: Expression applied to data fields to build the wanted value Must contain at least #nbData entries. If an entry is NULL or points to an empty string, the data returned are always default values.
in	<i>units</i>	: Wanted unit for each expression Must be NULL or contain at least #nbData entries. If NULL, no unit conversion is done. If an entry is NULL or points to an empty string, no unit conversion is applied to the data of the corresponding expression. When a unit conversion has to be applied, the result of the expression is considered to be the base unit (SI). For example if the wanted unit is gram/l, the unit of the expression is supposed to be kilogram/m3 (internally all data are converted to base unit of the actual fields unit which is coherent with the above assumption).
	<i>results</i>	[in/out] : Data read Must be a vector of at least #nbData pointers (entries) to values to read. If NULL, nothing is returned in results and sizes MUST be NULL (otherwise this is an error). An entry can be NULL, see #sizes for the actual behaviour.
	<i>sizes</i>	[in/out] : Number of allocated values in a #results entry. Must be a vector of at least #nbData integers. If NULL, results MUST also be NULL (otherwise this is an error). If a value is 0, nothing is returned. If a value is > 0, the corresponding entry in results must not be NULL and must have been allocated to be able to store as much float values as indicated. If a value is < 0, and the corresponding entry in results is NULL, the entry will be allocated with enough space to store the result and sizes modified to reflect the size of allocated data (may be more than actual used ones). If a value is < 0, and the corresponding entry in results is not NULL, this is an error.
out	<i>actualSize</i>	: Number of actual data needed to store result. It cannot be NULL. The actual number of values in the corresponding entry of #results are returned in this number (all entries need the same amount of result). If #result is NULL, the number of values which would be needed for each entry is returned.

in	<i>ignoreOutOfRange</i>	: Skip excess data. 0=false, other = true If true, #actualSize can be greater than any positive value of #sizes, if there is too much value to store they are ignored. If false, it generates an error. Has no effect on #sizes entries which are <= 0 (or if it is NULL).
in	<i>statistics</i>	: returns statistics on data instead of data themselves 0=false, other = true If statistics is true, ignoreOutOfRange must be false. And sizes must be <=0 or >=5. The returned values for each expression are: <ul style="list-style-type: none"> • Count of <i>valid</i> data taken into account. Invalid data are those which are equal to the default/missing value • Mean of the valid data. • Standard deviation of the valid data • Minimum value of the valid data • Maximum value of the valid data In this case actualSize always returns 5
in	<i>defaultValue</i>	: value to use for default/missing values This is the value you want to indicate that a value is missing or invalid.

Returns

BRATHL_SUCCESS (p. 17) or error code

References BRATHL_ERROR, and BRATHL_SUCCESS.

6 Class Documentation

6.1 `_structDateDSM` Struct Reference

```
#include <brathl.h>
```

Public Attributes

- `int32_t days`
- `int32_t muSeconds`
- `brathl_refDate refDate`
- `int32_t seconds`

6.1.1 Detailed Description

Day/seconds/microseconds date structure

6.1.2 Member Data Documentation

6.1.2.1 `int32_t _structDateDSM::days`

numbers of days

Referenced by `brathl_Julian2DSM()`, `brathl_Seconds2DSM()`, `brathl_YMDHMSM2DSM()`, and `brathl::CDate::SetDate()`.

6.1.2.2 `int32_t _structDateDSM::muSeconds`

numbers of microseconds

Referenced by `brathl_Julian2DSM()`, `brathl_Seconds2DSM()`, `brathl_YMDHMSM2DSM()`, and `brathl::CDate::SetDate()`.

6.1.2.3 `brathl_refDate _structDateDSM::refDate`

date reference (see **`brathl_refDate`** (p. 372))

Referenced by `brathl_Julian2DSM()`, `brathl_Seconds2DSM()`, `brathl_YMDHMSM2DSM()`, and `brathl::CDate::SetDate()`.

6.1.2.4 `int32_t _structDateDSM::seconds`

numbers of seconds

Referenced by `brathl_Julian2DSM()`, `brathl_Seconds2DSM()`, `brathl_YMDHMSM2DSM()`, and `brathl::CDate::SetDate()`.

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

- **`brathl.h`**

6.2 `_structDateJulian` Struct Reference

```
#include <brathl.h>
```

Public Attributes

- `double julian`

- **brathl_refDate refDate**

6.2.1 Detailed Description

Decimal julian date structure

6.2.2 Member Data Documentation

6.2.2.1 `double _structDateJulian::julian`

decimal julian day

Referenced by `brathl_DSM2Julian()`, `brathl_Seconds2Julian()`, `brathl_YMDHMSM2Julian()`, and `brathl::CDate::SetDate()`.

6.2.2.2 `brathl_refDate _structDateJulian::refDate`

date reference (see **brathl_refDate** (p. 372))

Referenced by `brathl_DSM2Julian()`, `brathl_Seconds2Julian()`, `brathl_YMDHMSM2Julian()`, and `brathl::CDate::SetDate()`.

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

- **brathl.h**

6.3 `_structDateSecond` Struct Reference

```
#include <brathl.h>
```

Public Attributes

- `double nbSeconds`
- **brathl_refDate refDate**

6.3.1 Detailed Description

Decimal seconds date structure

6.3.2 Member Data Documentation

6.3.2.1 `double _structDateSecond::nbSeconds`

numbers of seconds/microseconds

Referenced by `brathl_DSM2Seconds()`, `brathl_Julian2Seconds()`, `brathl_YMDHMSM2Seconds()`, and `brathl::CDate::SetDate()`.

6.3.2.2 `brathl_refDate _structDateSecond::refDate`

date reference (see **brathl_refDate** (p. 372))

Referenced by `brathl_DSM2Seconds()`, `brathl_Julian2Seconds()`, `brathl_YMDHMSM2Seconds()`, and `brathl::CDate::SetDate()`.

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

- **brathl.h**

6.4 `_structDateYMDHMSM` Struct Reference

```
#include <brathl.h>
```

Public Attributes

- `uint32_t day`
- `uint32_t hour`
- `uint32_t minute`
- `uint32_t month`
- `uint32_t muSecond`
- `uint32_t second`
- `uint32_t year`

6.4.1 Detailed Description

YYYY-MM-DD HH:MM:SS:MS date structure

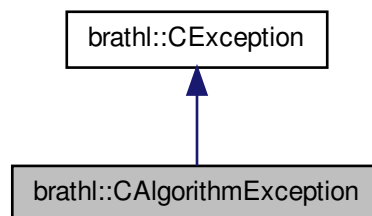
The documentation for this struct was generated from the following file:

- `brathl.h`

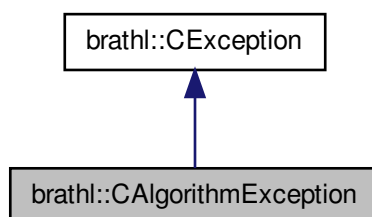
6.5 `brathl::CAgorithmException` Class Reference

```
#include <Exception.h>
```

Inheritance diagram for `brathl::CAgorithmException`:



Collaboration diagram for bratl::CAgorithmException:



Public Member Functions

- **CAgorithmException** ()
Empty CAgorithmException (p. 99) ctor.
- **CAgorithmException** (const string &message, int32_t errcode=**BRATHL_ERROR**)
- **CAgorithmException** (const string &message, const string &algorithmName, int32_t errcode)
- virtual const char * **TypeOf** () const
Identification of exception (human readable)
- virtual ~**CAgorithmException** () throw ()
Destructor.

Additional Inherited Members

6.5.1 Detailed Description

Algorithm Exception management class.

Version

1.0

6.5.2 Constructor & Destructor Documentation

6.5.2.1 bratl::CAgorithmException::CAgorithmException (const string & message, int32_t errcode = **BRATHL_ERROR**)
[inline]

Creates a new **CAgorithmException** (p. 99) object.

Parameters

<i>message</i>	[in] : error message
<i>errcode</i>	[in] : error code

6.5.2.2 bratl::CAgorithmException::CAgorithmException (const string & message, const string & algorithmName, int32_t errcode)

Creates a new **CAgorithmException** (p. 99) object.

Parameters

<i>message</i>	[in] : error message
<i>fileName</i>	[in] : file name in error
<i>errcode</i>	[in] : error code

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

- **Exception.h**
- Exception.cpp

6.6 brathl::CBratAlgoFilterGaussian1D Class Reference

```
#include <BratAlgoFilterGaussian1D.h>
```

Inherits brathl::CBratAlgoFilterGaussian.

Public Member Functions

- **CBratAlgoFilterGaussian1D** ()
- **CBratAlgoFilterGaussian1D** (const **CBratAlgoFilterGaussian1D** ©)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual uint32_t **GetDataWindowSize** ()
- virtual string **GetDescription** ()
- virtual string **GetName** ()
- **CBratAlgoFilterGaussian1D** & **operator=** (const **CBratAlgoFilterGaussian1D** ©)
- virtual double **Run** (CVectorBratAlgorithmParam &args)
- virtual ~**CBratAlgoFilterGaussian1D** ()

Protected Member Functions

- virtual void **CheckVarExpression** (uint32_t index)
- double **ComputeGaussian** ()
- void **Init** ()
- void **Set** (const **CBratAlgoFilterGaussian1D** ©)
- void **SetBeginOfFile** ()
- void **SetEndOfFile** ()
- virtual void **SetNextValues** ()
- virtual void **SetPreviousValues** (bool fromProduct)

Additional Inherited Members

6.6.1 Detailed Description

Algorithm base class.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 brathl::CBratAlgoFilterGaussian1D::CBratAlgoFilterGaussian1D ()

Default constructor

6.6.2.2 brathl::CBratAlgoFilterGaussian1D::CBratAlgoFilterGaussian1D (const CBratAlgoFilterGaussian1D & copy)

Copy constructor

6.6.2.3 virtual brathl::CBratAlgoFilterGaussian1D::~~CBratAlgoFilterGaussian1D () [inline],[virtual]

Destructor

6.6.3 Member Function Documentation

6.6.3.1 void brathl::CBratAlgoFilterGaussian1D::Dump (ostream & *fOut* = cerr) [virtual]

Dump function

Reimplemented from **brathl::CBratAlgorithmBase** (p. 123).

6.6.3.2 virtual string brathl::CBratAlgoFilterGaussian1D::GetDescription () [inline],[virtual]

Gets the description of the algorithm

Implements **brathl::CBratAlgorithmBase** (p. 124).

6.6.3.3 virtual string brathl::CBratAlgoFilterGaussian1D::GetName () [inline],[virtual]

Gets the name of the algorithm

Implements **brathl::CBratAlgorithmBase** (p. 124).

6.6.3.4 CBratAlgoFilterGaussian1D & brathl::CBratAlgoFilterGaussian1D::operator= (const CBratAlgoFilterGaussian1D & *copy*)

Overloads operator '='

6.6.3.5 double brathl::CBratAlgoFilterGaussian1D::Run (CVectorBratAlgorithmParam & *args*) [virtual]

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

Returns

the result of the execution

Implements **brathl::CBratAlgorithmBase** (p. 125).

References BRATHL_LOGIC_ERROR, and brathl::CTools::Format().

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

- BratAlgoFilterGaussian1D.h
- BratAlgoFilterGaussian1D.cpp

6.7 brathl::CBratAlgoFilterGaussian2D Class Reference

#include <BratAlgoFilterGaussian2D.h>

Inherits brathl::CBratAlgoFilterGaussian.

Public Member Functions

- **CBratAlgoFilterGaussian2D** ()

- **CBratAlgoFilterGaussian2D** (const **CBratAlgoFilterGaussian2D** ©)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual uint32_t **GetDataWindowSize** ()
- virtual string **GetDescription** ()
- virtual string **GetName** ()
- **CBratAlgoFilterGaussian2D** & **operator=** (const **CBratAlgoFilterGaussian2D** ©)
- virtual double **Run** (CVectorBratAlgorithmParam &args)
- virtual ~**CBratAlgoFilterGaussian2D** ()

Protected Member Functions

- void **CheckProduct** ()
- void **CheckVarExpression** (uint32_t index)
- virtual double **ComputeGaussian** (CExpressionValue &exprValue)
- double **ComputeMean** ()
- double **ComputeSingle** ()
- void **Init** ()
- virtual void **OpenProductFile** ()
- void **Set** (const **CBratAlgoFilterGaussian2D** ©)
- void **SetBeginOfFile** ()
- void **SetEndOfFile** ()

Additional Inherited Members

6.7.1 Detailed Description

Algorithm base class.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 bratl::CBratAlgoFilterGaussian2D::CBratAlgoFilterGaussian2D ()

Default constructor

6.7.2.2 bratl::CBratAlgoFilterGaussian2D::CBratAlgoFilterGaussian2D (const CBratAlgoFilterGaussian2D & copy)

Copy constructor

6.7.2.3 bratl::CBratAlgoFilterGaussian2D::~~CBratAlgoFilterGaussian2D () [virtual]

Destructor

6.7.3 Member Function Documentation

6.7.3.1 void bratl::CBratAlgoFilterGaussian2D::Dump (ostream & fOut = cerr) [virtual]

Dump function

Reimplemented from **bratl::CBratAlgorithmBase** (p. 123).

6.7.3.2 virtual string bratl::CBratAlgoFilterGaussian2D::GetDescription () [inline], [virtual]

Gets the description of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.7.3.3 virtual string brathl::CBratAlgoFilterGaussian2D::GetName () [inline],[virtual]

Gets the name of the algorithm

Implements **brathl::CBratAlgorithmBase** (p. 124).

6.7.3.4 **CBratAlgoFilterGaussian2D** & brathl::CBratAlgoFilterGaussian2D::operator= (const **CBratAlgoFilterGaussian2D** & copy)

Overloads operator '='

6.7.3.5 double brathl::CBratAlgoFilterGaussian2D::Run (CVectorBratAlgorithmParam & args) [virtual]

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

Returns

the result of the execution

Implements **brathl::CBratAlgorithmBase** (p. 125).

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

- BratAlgoFilterGaussian2D.h
- BratAlgoFilterGaussian2D.cpp

6.8 brathl::CBratAlgoFilterLanczos1D Class Reference

```
#include <BratAlgoFilterLanczos1D.h>
```

Inherits brathl::CBratAlgoFilterLanczos.

Public Member Functions

- **CBratAlgoFilterLanczos1D** ()
- **CBratAlgoFilterLanczos1D** (const **CBratAlgoFilterLanczos1D** ©)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual uint32_t **GetDataWindowSize** ()
- virtual string **GetDescription** ()
- virtual string **GetName** ()
- **CBratAlgoFilterLanczos1D** & operator= (const **CBratAlgoFilterLanczos1D** ©)
- virtual double **Run** (CVectorBratAlgorithmParam &args)
- virtual ~**CBratAlgoFilterLanczos1D** ()

Protected Member Functions

- virtual void **CheckVarExpression** (uint32_t index)
- double **ComputeLanczos** ()
- void **Init** ()
- void **Set** (const **CBratAlgoFilterLanczos1D** ©)
- void **SetBeginOfFile** ()
- void **SetEndOfFile** ()
- virtual void **SetNextValues** ()
- virtual void **SetPreviousValues** (bool fromProduct)

Additional Inherited Members

6.8.1 Detailed Description

Algorithm base class.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 bratl::CBratAlgoFilterLanczos1D::CBratAlgoFilterLanczos1D ()

Default constructor

6.8.2.2 bratl::CBratAlgoFilterLanczos1D::CBratAlgoFilterLanczos1D (const CBratAlgoFilterLanczos1D & copy)

Copy constructor

6.8.2.3 virtual bratl::CBratAlgoFilterLanczos1D::~CBratAlgoFilterLanczos1D () [inline], [virtual]

Destructor

6.8.3 Member Function Documentation

6.8.3.1 void bratl::CBratAlgoFilterLanczos1D::Dump (ostream & fOut = cerr) [virtual]

Dump function

Reimplemented from **bratl::CBratAlgorithmBase** (p. 123).

6.8.3.2 virtual string bratl::CBratAlgoFilterLanczos1D::GetDescription () [inline], [virtual]

Gets the description of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.8.3.3 virtual string bratl::CBratAlgoFilterLanczos1D::GetName () [inline], [virtual]

Gets the name of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.8.3.4 CBratAlgoFilterLanczos1D & bratl::CBratAlgoFilterLanczos1D::operator= (const CBratAlgoFilterLanczos1D & copy)

Overloads operator '='

6.8.3.5 double bratl::CBratAlgoFilterLanczos1D::Run (CVectorBratAlgorithmParam & args) [virtual]

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

Returns

the result of the execution

Implements **bratl::CBratAlgorithmBase** (p. 125).

References BRATHL_LOGIC_ERROR, and brathl::CTools::Format().

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

- BratAlgoFilterLanczos1D.h
- BratAlgoFilterLanczos1D.cpp

6.9 brathl::CBratAlgoFilterLanczos2D Class Reference

```
#include <BratAlgoFilterLanczos2D.h>
```

Inherits brathl::CBratAlgoFilterLanczos.

Public Member Functions

- **CBratAlgoFilterLanczos2D** ()
- **CBratAlgoFilterLanczos2D** (const **CBratAlgoFilterLanczos2D** ©)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual uint32_t **GetDataWindowSize** ()
- virtual string **GetDescription** ()
- virtual string **GetName** ()
- **CBratAlgoFilterLanczos2D** & **operator=** (const **CBratAlgoFilterLanczos2D** ©)
- virtual double **Run** (CVectorBratAlgorithmParam &args)
- virtual ~**CBratAlgoFilterLanczos2D** ()

Protected Member Functions

- void **CheckProduct** ()
- void **CheckVarExpression** (uint32_t index)
- virtual double **ComputeLanczos** (CExpressionValue &exprValue)
- double **ComputeMean** ()
- double **ComputeSingle** ()
- void **Init** ()
- virtual void **OpenProductFile** ()
- void **Set** (const **CBratAlgoFilterLanczos2D** ©)
- void **SetBeginOfFile** ()
- void **SetEndOfFile** ()

Additional Inherited Members

6.9.1 Detailed Description

Algorithm base class.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 brathl::CBratAlgoFilterLanczos2D::CBratAlgoFilterLanczos2D ()

Default constructor

6.9.2.2 brathl::CBratAlgoFilterLanczos2D::CBratAlgoFilterLanczos2D (const CBratAlgoFilterLanczos2D ©)

Copy constructor

6.9.2.3 `bratl::CBratAlgoFilterLanczos2D::~~CBratAlgoFilterLanczos2D ()` [virtual]

Destructor

6.9.3 Member Function Documentation

6.9.3.1 `void bratl::CBratAlgoFilterLanczos2D::Dump (ostream & fOut = cerr)` [virtual]

Dump function

Reimplemented from **bratl::CBratAlgorithmBase** (p. 123).

6.9.3.2 `virtual string bratl::CBratAlgoFilterLanczos2D::GetDescription ()` [inline],[virtual]

Gets the description of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.9.3.3 `virtual string bratl::CBratAlgoFilterLanczos2D::GetName ()` [inline],[virtual]

Gets the name of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.9.3.4 `CBratAlgoFilterLanczos2D & bratl::CBratAlgoFilterLanczos2D::operator= (const CBratAlgoFilterLanczos2D & copy)`

Overloads operator '='

6.9.3.5 `double bratl::CBratAlgoFilterLanczos2D::Run (CVectorBratAlgorithmParam & args)` [virtual]

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

Returns

the result of the execution

Implements **bratl::CBratAlgorithmBase** (p. 125).

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

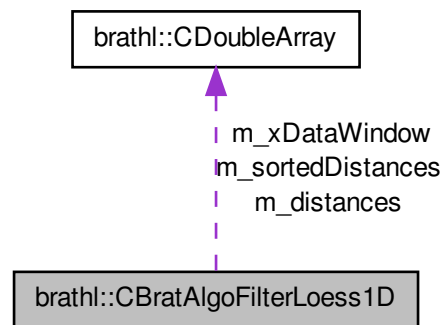
- BratAlgoFilterLanczos2D.h
- BratAlgoFilterLanczos2D.cpp

6.10 bratl::CBratAlgoFilterLoess1D Class Reference

```
#include <BratAlgoFilterLoess1D.h>
```

Inherits **bratl::CBratAlgoFilterLoess**.

Collaboration diagram for bratl::CBratAlgoFilterLoess1D:



Public Member Functions

- **CBratAlgoFilterLoess1D** ()
- **CBratAlgoFilterLoess1D** (const **CBratAlgoFilterLoess1D** ©)
- virtual void **CheckInputParams** (CVectorBratAlgorithmParam &args)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual uint32_t **GetDataWindowSize** ()
- virtual string **GetDescription** ()
- virtual string **GetInputParamDesc** (uint32_t indexParam)
- virtual
CBratAlgorithmParam::bratAlgoParamTypeVal **GetInputParamFormat** (uint32_t indexParam)
- virtual string **GetInputParamUnit** (uint32_t indexParam)
- virtual string **GetName** ()
- virtual uint32_t **GetNumInputParam** ()
- virtual string **GetOutputUnit** ()
- virtual double **GetParamDefaultValue** (uint32_t indexParam)
- virtual string **GetParamName** (uint32_t indexParam)
- **CBratAlgoFilterLoess1D** & **operator=** (const **CBratAlgoFilterLoess1D** ©)
- virtual double **Run** (CVectorBratAlgorithmParam &args)
- virtual void **SetParamValues** (CVectorBratAlgorithmParam &args)
- virtual ~**CBratAlgoFilterLoess1D** ()

Protected Member Functions

- double **ApplyFilter** ()
- virtual void **CheckVarExpression** (uint32_t index)
- double **ComputeLoess** ()
- void **FitLinearEst** (const double x, const double c0, const double c1, const double cov00, const double cov01, const double cov11, double *y, double *y_err)
- void **FitWLinear** (const double *x, const uint32_t xstride, const double *w, const uint32_t wstride, const double *y, const uint32_t ystride, const uint32_t n, double *c0, double *c1, double *cov_00, double *cov_01, double *cov_11, double *chisq)
- void **Init** ()
- virtual void **InsertCurrentValueDataWindow1D** ()
- virtual void **RemoveFirstItemDataWindow1D** ()

- void **Set** (const **CBratAlgoFilterLoess1D** ©)
- void **SetBeginOfFile** ()
- void **SetEndOfFile** ()
- virtual void **SetNextValues** ()
- virtual void **SetPreviousValues** (bool fromProduct)
- virtual void **TreatLeftEdge1D** (uint32_t shiftSymmetry, uint32_t index)
- virtual void **TreatRightEdge1D** (uint32_t shiftSymmetry, uint32_t index)
- double **Tricube** (double u, double t)

Protected Attributes

- **CDoubleArray** **m_distances**
- **CDoubleArray** **m_sortedDistances**
- **CDoubleArray** **m_xDataWindow**
- double **m_xValue**
- double **m_xValueNext**
- double **m_xValuePrev**

Static Protected Attributes

- static const uint32_t **m_EXTRAPOLATE_PARAM_INDEX**
- static const uint32_t **m_INPUT_PARAMS** = 4
- static const uint32_t **m_VALID_PARAM_INDEX** = 3
- static const uint32_t **m_WINDOW_PARAM_INDEX** = 2
- static const uint32_t **m_X_PARAM_INDEX** = 1

Additional Inherited Members

6.10.1 Detailed Description

Algorithm base class.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 bratl::CBratAlgoFilterLoess1D::CBratAlgoFilterLoess1D ()

Default constructor

6.10.2.2 bratl::CBratAlgoFilterLoess1D::CBratAlgoFilterLoess1D (const CBratAlgoFilterLoess1D ©)

Copy constructor

6.10.2.3 virtual bratl::CBratAlgoFilterLoess1D::~~CBratAlgoFilterLoess1D () [inline], [virtual]

Destructor

6.10.3 Member Function Documentation

6.10.3.1 void bratl::CBratAlgoFilterLoess1D::Dump (ostream & fOut = cerr) [virtual]

Dump function

Reimplemented from **bratl::CBratAlgorithmBase** (p. 123).

6.10.3.2 `virtual string bratl::CBratAlgoFilterLoess1D::GetDescription () [inline],[virtual]`

Gets the description of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.10.3.3 `virtual string bratl::CBratAlgoFilterLoess1D::GetInputParamDesc (uint32_t indexParam) [inline],[virtual]`

Gets the description of an input parameter.

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References `bratl::CTools::Format()`.

6.10.3.4 `virtual CBratAlgorithmParam::bratAlgoParamTypeVal bratl::CBratAlgoFilterLoess1D::GetInputParamFormat (uint32_t indexParam) [inline],[virtual]`

Gets the format of an input parameter : `CBratAlgorithmParam::T_DOUBLE` for double `CBratAlgorithmParam::T_FLOAT` for float `CBratAlgorithmParam::T_INT` for integer `CBratAlgorithmParam::T_LONG` for long integer `CBratAlgorithmParam::T_STRING` for string `CBratAlgorithmParam::T_CHAR` for a character

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References `bratl::CTools::Format()`.

6.10.3.5 `virtual string bratl::CBratAlgoFilterLoess1D::GetInputParamUnit (uint32_t indexParam) [inline],[virtual]`

Gets the unit of an input parameter :

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **bratl::CBratAlgorithmBase** (p. 124).

References `bratl::CTools::Format()`.

6.10.3.6 `virtual string bratl::CBratAlgoFilterLoess1D::GetName () [inline],[virtual]`

Gets the name of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.10.3.7 `virtual uint32_t bratl::CBratAlgoFilterLoess1D::GetNumInputParam () [inline],[virtual]`

Gets the number of input parameters to pass to the 'Run' function

Implements **bratl::CBratAlgorithmBase** (p. 125).

6.10.3.8 `virtual string bratl::CBratAlgoFilterLoess1D::GetOutputUnit () [inline],[virtual]`

Gets the unit of an output value returned by the 'Run' function.

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **brathl::CBratAlgorithmBase** (p. 125).

6.10.3.9 CBratAlgoFilterLoess1D & brathl::CBratAlgoFilterLoess1D::operator= (const CBratAlgoFilterLoess1D & copy)

Overloads operator '='

6.10.3.10 double brathl::CBratAlgoFilterLoess1D::Run (CVectorBratAlgorithmParam & args) [virtual]

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

Returns

the result of the execution

Implements **brathl::CBratAlgorithmBase** (p. 125).

References BRATHL_LOGIC_ERROR, and brathl::CTools::Format().

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

- BratAlgoFilterLoess1D.h
- BratAlgoFilterLoess1D.cpp

6.11 brathl::CBratAlgoFilterLoess2D Class Reference

```
#include <BratAlgoFilterLoess2D.h>
```

Inherits brathl::CBratAlgoFilterLoess.

Public Member Functions

- **CBratAlgoFilterLoess2D** ()
- **CBratAlgoFilterLoess2D** (const **CBratAlgoFilterLoess2D** ©)
- virtual void **CheckInputParams** (CVectorBratAlgorithmParam &args)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual uint32_t **GetDataWindowSize** ()
- virtual string **GetDescription** ()
- virtual string **GetInputParamDesc** (uint32_t indexParam)
- virtual
CBratAlgorithmParam::bratAlgoParamTypeVal **GetInputParamFormat** (uint32_t indexParam)
- virtual string **GetInputParamUnit** (uint32_t indexParam)
- virtual string **GetName** ()
- virtual uint32_t **GetNumInputParam** ()
- virtual string **GetOutputUnit** ()
- virtual double **GetParamDefaultValue** (uint32_t indexParam)
- virtual string **GetParamName** (uint32_t indexParam)
- **CBratAlgoFilterLoess2D** & **operator=** (const **CBratAlgoFilterLoess2D** ©)
- virtual double **Run** (CVectorBratAlgorithmParam &args)
- virtual void **SetParamValues** (CVectorBratAlgorithmParam &args)
- virtual ~**CBratAlgoFilterLoess2D** ()

Protected Member Functions

- double **ApplyFilter** ()
- void **CheckProduct** ()
- void **CheckVarExpression** (uint32_t index)
- void **ComputeInitialWeights** ()
- double **ComputeLoess** ()
- double **ComputeMean** ()
- double **ComputeSingle** ()
- void **Init** ()
- virtual void **OpenProductFile** ()
- void **PrepareDataValues** ()
- void **PrepareDataWindow** ()
- void **Set** (const **CBratAlgoFilterLoess2D** ©)
- void **SetBeginOfFile** ()
- void **SetEndOfFile** ()

Static Protected Attributes

- static const uint32_t **m_EXTRAPOLATE_PARAM_INDEX** = 4
- static const uint32_t **m_INPUT_PARAMS** = 5
- static const uint32_t **m_VALID_PARAM_INDEX** = 3
- static const uint32_t **m_WINDOW_HEIGHT_PARAM_INDEX** = 2
- static const uint32_t **m_WINDOW_WIDTH_PARAM_INDEX** = 1

Additional Inherited Members

6.11.1 Detailed Description

Algorithm base class.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 bratl::CBratAlgoFilterLoess2D::CBratAlgoFilterLoess2D ()

Default constructor

6.11.2.2 bratl::CBratAlgoFilterLoess2D::CBratAlgoFilterLoess2D (const **CBratAlgoFilterLoess2D** ©)

Copy constructor

6.11.2.3 bratl::CBratAlgoFilterLoess2D::~~CBratAlgoFilterLoess2D () [virtual]

Destructor

6.11.3 Member Function Documentation

6.11.3.1 void bratl::CBratAlgoFilterLoess2D::Dump (ostream & fOut = cerr) [virtual]

Dump function

Reimplemented from **bratl::CBratAlgorithmBase** (p. 123).

6.11.3.2 `virtual string bratl::CBratAlgoFilterLoess2D::GetDescription () [inline],[virtual]`

Gets the description of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.11.3.3 `virtual string bratl::CBratAlgoFilterLoess2D::GetInputParamDesc (uint32_t indexParam) [inline],[virtual]`

Gets the description of an input parameter.

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References `bratl::CTools::Format()`.

6.11.3.4 `virtual CBratAlgorithmParam::bratAlgoParamTypeVal bratl::CBratAlgoFilterLoess2D::GetInputParamFormat (uint32_t indexParam) [inline],[virtual]`

Gets the format of an input parameter : `CBratAlgorithmParam::T_DOUBLE` for double `CBratAlgorithmParam::T_FLOAT` for float `CBratAlgorithmParam::T_INT` for integer `CBratAlgorithmParam::T_LONG` for long integer `CBratAlgorithmParam::T_STRING` for string `CBratAlgorithmParam::T_CHAR` for a character

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References `bratl::CTools::Format()`.

6.11.3.5 `virtual string bratl::CBratAlgoFilterLoess2D::GetInputParamUnit (uint32_t indexParam) [inline],[virtual]`

Gets the unit of an input parameter :

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **bratl::CBratAlgorithmBase** (p. 124).

References `bratl::CTools::Format()`.

6.11.3.6 `virtual string bratl::CBratAlgoFilterLoess2D::GetName () [inline],[virtual]`

Gets the name of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.11.3.7 `virtual uint32_t bratl::CBratAlgoFilterLoess2D::GetNumInputParam () [inline],[virtual]`

Gets the number of input parameters to pass to the 'Run' function

Implements **bratl::CBratAlgorithmBase** (p. 125).

6.11.3.8 `virtual string bratl::CBratAlgoFilterLoess2D::GetOutputUnit () [inline],[virtual]`

Gets the unit of an output value returned by the 'Run' function.

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **bratl::CBratAlgorithmBase** (p. 125).

6.11.3.9 CBratAlgoFilterLoess2D & bratl::CBratAlgoFilterLoess2D::operator= (const CBratAlgoFilterLoess2D & copy)

Overloads operator '='

6.11.3.10 double bratl::CBratAlgoFilterLoess2D::Run (CVectorBratAlgorithmParam & args) [virtual]

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

Returns

the result of the execution

Implements **bratl::CBratAlgorithmBase** (p. 125).

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

- BratAlgoFilterLoess2D.h
- BratAlgoFilterLoess2D.cpp

6.12 bratl::CBratAlgoFilterMedian1D Class Reference

```
#include <BratAlgoFilterMedian1D.h>
```

Inherits **bratl::CBratAlgoFilterMedian**.

Public Member Functions

- **CBratAlgoFilterMedian1D** ()
- **CBratAlgoFilterMedian1D** (const **CBratAlgoFilterMedian1D** ©)
- virtual void **CheckInputParams** (CVectorBratAlgorithmParam &args)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual uint32_t **GetDataWindowSize** ()
- virtual string **GetDescription** ()
- virtual string **GetInputParamDesc** (uint32_t indexParam)
- virtual
CBratAlgorithmParam::bratAlgoParamTypeVal **GetInputParamFormat** (uint32_t indexParam)
- virtual string **GetInputParamUnit** (uint32_t indexParam)
- virtual string **GetName** ()
- virtual uint32_t **GetNumInputParam** ()
- virtual string **GetOutputUnit** ()
- virtual double **GetParamDefaultValue** (uint32_t indexParam)
- virtual string **GetParamName** (uint32_t indexParam)
- **CBratAlgoFilterMedian1D** & **operator=** (const **CBratAlgoFilterMedian1D** ©)
- virtual double **Run** (CVectorBratAlgorithmParam &args)
- virtual void **SetParamValues** (CVectorBratAlgorithmParam &args)
- virtual ~**CBratAlgoFilterMedian1D** ()

Protected Member Functions

- virtual void **CheckVarExpression** (uint32_t index)
- void **Init** ()
- void **Set** (const **CBratAlgoFilterMedian1D** ©)
- void **SetBeginOfFile** ()
- void **SetEndOfFile** ()
- virtual void **SetNextValues** ()
- virtual void **SetPreviousValues** (bool fromProduct)

Static Protected Attributes

- static const uint32_t **m_EXTRAPOLATE_PARAM_INDEX** = 3
- static const uint32_t **m_INPUT_PARAMS** = 4
- static const uint32_t **m_VALID_PARAM_INDEX** = 2
- static const uint32_t **m_WINDOW_PARAM_INDEX** = 1

Additional Inherited Members

6.12.1 Detailed Description

Algorithm base class.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 brathl::CBratAlgoFilterMedian1D::CBratAlgoFilterMedian1D ()

Default constructor

6.12.2.2 brathl::CBratAlgoFilterMedian1D::CBratAlgoFilterMedian1D (const **CBratAlgoFilterMedian1D** ©)

Copy constructor

6.12.2.3 virtual brathl::CBratAlgoFilterMedian1D::~CBratAlgoFilterMedian1D () [inline], [virtual]

Destructor

6.12.3 Member Function Documentation

6.12.3.1 void brathl::CBratAlgoFilterMedian1D::Dump (ostream & fOut = cerr) [virtual]

Dump function

Reimplemented from **brathl::CBratAlgorithmBase** (p. 123).

6.12.3.2 virtual string brathl::CBratAlgoFilterMedian1D::GetDescription () [inline], [virtual]

Gets the description of the algorithm

Implements **brathl::CBratAlgorithmBase** (p. 124).

6.12.3.3 virtual string brathl::CBratAlgoFilterMedian1D::GetInputParamDesc (uint32_t indexParam) [inline], [virtual]

Gets the description of an input parameter.

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References bratl::CTools::Format().

6.12.3.4 virtual CBratAlgorithmParam::bratAlgoParamTypeVal bratl::CBratAlgoFilterMedian1D::GetInputParamFormat (uint32_t *indexParam*) [inline],[virtual]

Gets the format of an input parameter : CBratAlgorithmParam::T_DOUBLE for double CBratAlgorithmParam::T_FLOAT for float CBratAlgorithmParam::T_INT for integer CBratAlgorithmParam::T_LONG for long integer CBratAlgorithmParam::T_STRING for string CBratAlgorithmParam::T_CHAR for a character

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References bratl::CTools::Format().

6.12.3.5 virtual string bratl::CBratAlgoFilterMedian1D::GetInputParamUnit (uint32_t *indexParam*) [inline],[virtual]

Gets the unit of an input parameter :

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **bratl::CBratAlgorithmBase** (p. 124).

References bratl::CTools::Format().

6.12.3.6 virtual string bratl::CBratAlgoFilterMedian1D::GetName () [inline],[virtual]

Gets the name of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.12.3.7 virtual uint32_t bratl::CBratAlgoFilterMedian1D::GetNumInputParam () [inline],[virtual]

Gets the number of input parameters to pass to the 'Run' function

Implements **bratl::CBratAlgorithmBase** (p. 125).

6.12.3.8 virtual string bratl::CBratAlgoFilterMedian1D::GetOutputUnit () [inline],[virtual]

Gets the unit of an output value returned by the 'Run' function.

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **bratl::CBratAlgorithmBase** (p. 125).

6.12.3.9 CBratAlgoFilterMedian1D & bratl::CBratAlgoFilterMedian1D::operator= (const CBratAlgoFilterMedian1D & *copy*)

Overloads operator '='

6.12.3.10 `double brathl::CBratAlgoFilterMedian1D::Run (CVectorBratAlgorithmParam & args) [virtual]`

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

Returns

the result of the execution

Implements **brathl::CBratAlgorithmBase** (p. 125).

References BRATHL_LOGIC_ERROR, and brathl::CTools::Format().

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

- BratAlgoFilterMedian1D.h
- BratAlgoFilterMedian1D.cpp

6.13 brathl::CBratAlgoFilterMedian2D Class Reference

`#include <BratAlgoFilterMedian2D.h>`

Inherits brathl::CBratAlgoFilterMedian.

Public Member Functions

- **CBratAlgoFilterMedian2D** ()
- **CBratAlgoFilterMedian2D** (const **CBratAlgoFilterMedian2D** ©)
- virtual void **CheckInputParams** (CVectorBratAlgorithmParam &args)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual uint32_t **GetDataWindowSize** ()
- virtual string **GetDescription** ()
- virtual string **GetInputParamDesc** (uint32_t indexParam)
- virtual
CBratAlgorithmParam::bratAlgoParamTypeVal **GetInputParamFormat** (uint32_t indexParam)
- virtual string **GetInputParamUnit** (uint32_t indexParam)
- virtual string **GetName** ()
- virtual uint32_t **GetNumInputParam** ()
- virtual string **GetOutputUnit** ()
- virtual double **GetParamDefaultValue** (uint32_t indexParam)
- virtual string **GetParamName** (uint32_t indexParam)
- **CBratAlgoFilterMedian2D** & **operator=** (const **CBratAlgoFilterMedian2D** ©)
- virtual double **Run** (CVectorBratAlgorithmParam &args)
- virtual void **SetParamValues** (CVectorBratAlgorithmParam &args)
- virtual ~**CBratAlgoFilterMedian2D** ()

Protected Member Functions

- void **CheckProduct** ()
- void **CheckVarExpression** (uint32_t index)
- double **ComputeMean** ()
- double **ComputeSingle** ()
- void **Init** ()
- virtual void **OpenProductFile** ()
- void **PrepareDataValues** ()
- void **PrepareDataWindow** ()
- void **Set** (const **CBratAlgoFilterMedian2D** ©)
- void **SetBeginOfFile** ()
- void **SetEndOfFile** ()

Static Protected Attributes

- static const uint32_t **m_EXTRAPOLATE_PARAM_INDEX** = 4
- static const uint32_t **m_INPUT_PARAMS** = 5
- static const uint32_t **m_VALID_PARAM_INDEX** = 3
- static const uint32_t **m_WINDOW_HEIGHT_PARAM_INDEX** = 2
- static const uint32_t **m_WINDOW_WIDTH_PARAM_INDEX** = 1

Additional Inherited Members

6.13.1 Detailed Description

Algorithm base class.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 bratl::CBratAlgoFilterMedian2D::CBratAlgoFilterMedian2D ()

Default constructor

6.13.2.2 bratl::CBratAlgoFilterMedian2D::CBratAlgoFilterMedian2D (const **CBratAlgoFilterMedian2D** ©)

Copy constructor

6.13.2.3 bratl::CBratAlgoFilterMedian2D::~~CBratAlgoFilterMedian2D () [virtual]

Destructor

6.13.3 Member Function Documentation

6.13.3.1 void bratl::CBratAlgoFilterMedian2D::Dump (ostream & fOut = cerr) [virtual]

Dump function

Reimplemented from **bratl::CBratAlgorithmBase** (p. 123).

6.13.3.2 virtual string bratl::CBratAlgoFilterMedian2D::GetDescription () [inline], [virtual]

Gets the description of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.13.3.3 `virtual string bratl::CBratAlgoFilterMedian2D::GetInputParamDesc (uint32_t indexParam) [inline], [virtual]`

Gets the description of an input parameter.

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References **bratl::CTools::Format()**.

6.13.3.4 `virtual CBratAlgorithmParam::bratAlgoParamTypeVal bratl::CBratAlgoFilterMedian2D::GetInputParamFormat (uint32_t indexParam) [inline], [virtual]`

Gets the format of an input parameter : CBratAlgorithmParam::T_DOUBLE for double CBratAlgorithmParam::T_FLOAT for float CBratAlgorithmParam::T_INT for integer CBratAlgorithmParam::T_LONG for long integer CBratAlgorithmParam::T_STRING for string CBratAlgorithmParam::T_CHAR for a character

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References **bratl::CTools::Format()**.

6.13.3.5 `virtual string bratl::CBratAlgoFilterMedian2D::GetInputParamUnit (uint32_t indexParam) [inline], [virtual]`

Gets the unit of an input parameter :

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **bratl::CBratAlgorithmBase** (p. 124).

References **bratl::CTools::Format()**.

6.13.3.6 `virtual string bratl::CBratAlgoFilterMedian2D::GetName () [inline], [virtual]`

Gets the name of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.13.3.7 `virtual uint32_t bratl::CBratAlgoFilterMedian2D::GetNumInputParam () [inline], [virtual]`

Gets the number of input parameters to pass to the 'Run' function

Implements **bratl::CBratAlgorithmBase** (p. 125).

6.13.3.8 `virtual string bratl::CBratAlgoFilterMedian2D::GetOutputUnit () [inline], [virtual]`

Gets the unit of an output value returned by the 'Run' function.

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **bratl::CBratAlgorithmBase** (p. 125).

6.13.3.9 CBratAlgoFilterMedian2D & bratl::CBratAlgoFilterMedian2D::operator= (const CBratAlgoFilterMedian2D & copy)

Overloads operator '='

6.13.3.10 double bratl::CBratAlgoFilterMedian2D::Run (CVectorBratAlgorithmParam & args) [virtual]

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

Returns

the result of the execution

Implements **bratl::CBratAlgorithmBase** (p. 125).

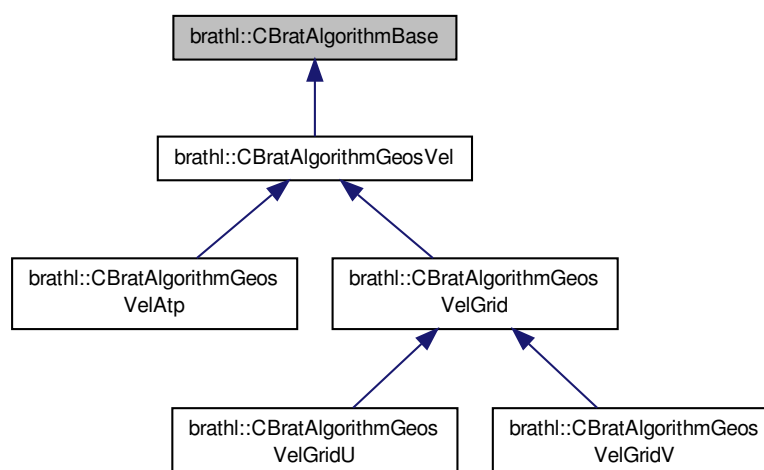
The documentation for this class was generated from the following files:

- BratAlgoFilterMedian2D.h
- BratAlgoFilterMedian2D.cpp

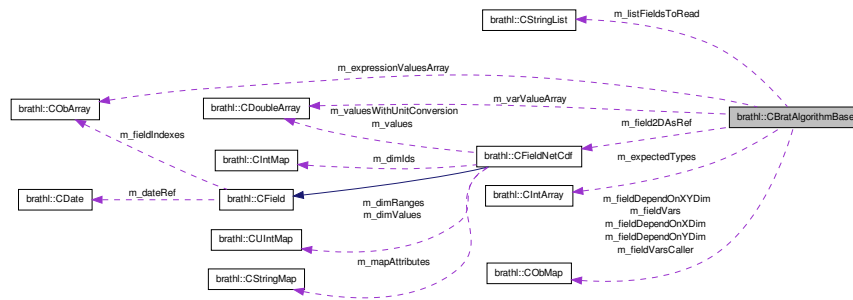
6.14 bratl::CBratAlgorithmBase Class Reference

```
#include <BratAlgorithmBase.h>
```

Inheritance diagram for bratl::CBratAlgorithmBase:



Collaboration diagram for brathl::CBratAlgorithmBase:



Public Member Functions

- **CBratAlgorithmBase** ()
- **CBratAlgorithmBase** (const **CBratAlgorithmBase** &o)
- void **CheckConstantParam** (uint32_t indexParam)
- virtual void **CheckInputParams** (CVectorBratAlgorithmParam &args)
- virtual void **CheckInputTypeParams** (uint32_t index, CBratAlgorithmParam::bratAlgoParamTypeVal expectedType, CVectorBratAlgorithmParam &args)
- virtual void **CheckInputTypeParams** (uint32_t index, const **CIntArray** &expectedTypes, CVectorBratAlgorithmParam &args)
- virtual void **Dump** (ostream &fOut=cerr)
- string **GetAlgoExpression** ()
- **CObArray** * **GetAlgoParamExpressions** ()
- virtual string **GetDescription** ()=0
- virtual string **GetInputParamDesc** (uint32_t indexParam)=0
- string **GetInputParamDescWithDefValueLabel** (uint32_t indexParam)
- virtual
CBratAlgorithmParam::bratAlgoParamTypeVal **GetInputParamFormat** (uint32_t indexParam)=0
- virtual string **GetInputParamFormatAsString** (uint32_t indexParam)
- virtual string **GetInputParamUnit** (uint32_t indexParam)=0
- virtual string **GetName** ()=0
- virtual uint32_t **GetNumInputParam** ()=0
- virtual string **GetOutputUnit** ()=0
- virtual double **GetParamDefaultValue** (uint32_t)
- void **GetParamDefValue** (uint32_t indexParam, double &value)
- void **GetParamDefValue** (uint32_t indexParam, float &value)
- void **GetParamDefValue** (uint32_t indexParam, uint32_t &value)
- void **GetParamDefValue** (uint32_t indexParam, uint64_t &value)
- void **GetParamDefValue** (uint32_t indexParam, int32_t &value)
- void **GetParamDefValue** (uint32_t indexParam, int64_t &value)
- string **GetParamDefValueAsLabel** (uint32_t indexParam)
- string **GetParamDefValueAsString** (uint32_t indexParam)
- virtual string **GetParamName** (uint32_t)
- **CProductNetCdf** * **GetProductNetCdf** (CProduct *product)
- string **GetSyntax** ()
- **CBratAlgorithmBase** & **operator=** (const **CBratAlgorithmBase** &o)
- virtual double **Run** (CVectorBratAlgorithmParam &args)=0
- void **SetAlgoExpression** (const string &value)
- void **SetAlgoParamExpressions** (const CStringArray &values)
- void **SetAlgoParamExpressions** (const **CObArray** &obArray)
- virtual void **SetProduct** (CProduct *product, bool forceReplace=false)
- virtual ~**CBratAlgorithmBase** ()

Static Public Member Functions

- static double **ExecInternal** (CBratAlgorithmBase *algo, CVectorBratAlgorithmParam &arg)
- static CBratAlgorithmBase * **GetNew** (const char *algorithmName)
- static void **RegisterAlgorithms** ()

Protected Member Functions

- void **AddXOrYFieldDependency** (CFieldNetCdf *field, CFieldNetCdf *field2DAsRef)
- void **AddXOrYFieldDependency** (CFieldNetCdf *field, const string &xDimName, const string &yDimName)
- virtual void **CheckComplexExpression** (uint32_t index)
- virtual void **CheckVarExpression2D** (uint32_t index)
- virtual void **DeleteExpressionValuesArray** ()
- virtual void **DeleteFieldNetCdf** ()
- virtual void **DeleteProduct** ()
- virtual void **GetAllData** (CExpression *expression, CDoubleArray &data)
- virtual void **GetData1D** (int32_t iRecord)
- CObArray * **GetDataExpressionValues** (uint32_t indexExpr)
- double **GetDataValue** (uint32_t indexExpr)
- double **GetDataValue** (uint32_t indexExpr, uint32_t x)
- double **GetDataValue** (uint32_t indexExpr, uint32_t x, uint32_t y)
- void **GetExpressionDataValuesAsArrayOfSingleValue** (uint32_t indexExpr, double *&values, uint32_t &nbValues)
- CFieldNetCdf * **GetField2DAsRef** ()
- virtual void **GetNextData** ()
- void **Init** ()
- void **InitComplexExpressionArray** ()
- virtual void **NewExpressionValuesArray** ()
- virtual void **OpenProductFile** ()
- virtual void **OpenProductFile** (CProduct *product)
- virtual void **PrepareDataValues2DComplexExpression** (CExpressionValue &exprValue, uint32_t algoExprIndex)
- virtual void **PrepareDataValues2DComplexExpressionWithAlgo** (CExpressionValue &exprValue, uint32_t algoExprIndex)
- virtual void **PrepareDataValues2DOneField** (CExpressionValue &exprValue, uint32_t algoExprIndex)
- virtual void **ProcessOpeningProductNetCdf** ()
- virtual void **ProcessOpeningProductNetCdf** (CProduct *product)
- virtual uint32_t **ReadProductData** (int32_t iRecord)
- virtual uint32_t **ReadProductData** (int32_t iRecord, CExpression *expression)
- virtual uint32_t **ReadProductData** (int32_t iRecord, const CObArrayOb &algoParamExpressions)
- virtual uint32_t **ReadProductData** (CProduct *product, int32_t iRecord, const CObArrayOb &arrayExpressions)
- void **Set** (const CBratAlgorithmBase &o)
- virtual void **SetBeginOfFile** ()
- virtual void **SetEndOfFile** ()
- void **SetField2DAsRef** ()
- virtual void **SetNextValues** ()
- virtual void **SetPreviousValues** (bool fromProduct)

Protected Attributes

- string **m_algoExpression**
- CObArrayOb **m_algoParamExpressions**
- CProduct * **m_callerProduct**
- int32_t **m_callerProductRecordPrev**
- string **m_currentFileName**
- CIntArray **m_expectedTypes**
- CObArray * **m_expressionValuesArray**
- CFieldNetCdf * **m_field2DAsRef**
- CObMap **m_fieldDependOnXDim**
- CObMap **m_fieldDependOnXYDim**
- CObMap **m_fieldDependOnYDim**
- CObMap **m_fieldVars**
- CObMap **m_fieldVarsCaller**
- int32_t **m_indexRecordToRead**
- vector< bool > **m_isComplexExpression**
- vector< bool > **m_isComplexExpressionWithAlgo**
- CStringList **m_listFieldsToRead**
- int32_t **m_nProductRecords**
- CProduct * **m_product**
- CDoubleArray * **m_varValueArray**

Static Protected Attributes

- static bool **m_algorithmsRegistered** = false

6.14.1 Detailed Description

Algorithm base class.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 bratl::CBratAlgorithmBase::CBratAlgorithmBase ()

Default constructor

6.14.2.2 bratl::CBratAlgorithmBase::CBratAlgorithmBase (const CBratAlgorithmBase & o)

Copy constructor

6.14.2.3 bratl::CBratAlgorithmBase::~~CBratAlgorithmBase () [virtual]

Destructor

6.14.3 Member Function Documentation

6.14.3.1 void bratl::CBratAlgorithmBase::Dump (ostream & fOut = cerr) [virtual]

Dump function

Reimplemented in **bratl::CBratAlgorithmGeosVelGridV** (p. 23), **bratl::CBratAlgorithmGeosVelGridU** (p. 23), **bratl::CBratAlgoFilterLoess1D** (p. 109), **bratl::CBratAlgoFilterLoess2D** (p. 112), **bratl::CBratAlgoFilterMedian2D** (p. 118), **bratl::CBratAlgoFilterMedian1D** (p. 115), **bratl::CBratAlgorithmGeosVelGrid** (p. 23),

bratl::CBratAlgorithmGeosVelAtp (p. 130), **bratl::CBratAlgoFilterGaussian2D** (p. 103), **bratl::CBratAlgoFilterLanczos2D** (p. 107), **bratl::CBratAlgoFilterGaussian1D** (p. 102), **bratl::CBratAlgoFilterLanczos1D** (p. 105), and **bratl::CBratAlgorithmGeosVel** (p. 128).

Referenced by `bratl::CBratAlgorithmGeosVel::Dump()`.

6.14.3.2 `virtual string bratl::CBratAlgorithmBase::GetDescription ()` [pure virtual]

Gets the description of the algorithm

Implemented in **bratl::CBratAlgorithmGeosVelGridV** (p. 23), **bratl::CBratAlgorithmGeosVelGridU** (p. 23), **bratl::CBratAlgoFilterGaussian1D** (p. 102), **bratl::CBratAlgoFilterGaussian2D** (p. 103), **bratl::CBratAlgoFilterLanczos1D** (p. 105), **bratl::CBratAlgoFilterLanczos2D** (p. 107), **bratl::CBratAlgoFilterLoess1D** (p. 110), **bratl::CBratAlgoFilterLoess2D** (p. 113), **bratl::CBratAlgoFilterMedian1D** (p. 115), **bratl::CBratAlgoFilterMedian2D** (p. 118), and **bratl::CBratAlgorithmGeosVelAtp** (p. 130).

6.14.3.3 `virtual string bratl::CBratAlgorithmBase::GetInputParamDesc (uint32_t indexParam)` [pure virtual]

Gets the description of an input parameter.

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implemented in **bratl::CBratAlgoFilterLoess1D** (p. 110), **bratl::CBratAlgoFilterLoess2D** (p. 113), **bratl::CBratAlgoFilterMedian1D** (p. 115), **bratl::CBratAlgoFilterMedian2D** (p. 119), **bratl::CBratAlgorithmGeosVelAtp** (p. 130), and **bratl::CBratAlgorithmGeosVelGrid** (p. 23).

6.14.3.4 `virtual CBratAlgorithmParam::bratAlgoParamTypeVal bratl::CBratAlgorithmBase::GetInputParamFormat (uint32_t indexParam)` [pure virtual]

Gets the format of an input parameter : `CBratAlgorithmParam::T_DOUBLE` for double `CBratAlgorithmParam::T_FLOAT` for float `CBratAlgorithmParam::T_INT` for integer `CBratAlgorithmParam::T_LONG` for long integer `CBratAlgorithmParam::T_STRING` for string `CBratAlgorithmParam::T_CHAR` for a character

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implemented in **bratl::CBratAlgoFilterLoess1D** (p. 110), **bratl::CBratAlgoFilterLoess2D** (p. 113), **bratl::CBratAlgoFilterMedian2D** (p. 119), **bratl::CBratAlgoFilterMedian1D** (p. 116), **bratl::CBratAlgorithmGeosVelAtp** (p. 131), and **bratl::CBratAlgorithmGeosVelGrid** (p. 23).

6.14.3.5 `virtual string bratl::CBratAlgorithmBase::GetInputParamUnit (uint32_t indexParam)` [pure virtual]

Gets the unit of an input parameter :

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implemented in **bratl::CBratAlgoFilterLoess1D** (p. 110), **bratl::CBratAlgoFilterLoess2D** (p. 113), **bratl::CBratAlgoFilterMedian2D** (p. 119), **bratl::CBratAlgoFilterMedian1D** (p. 116), **bratl::CBratAlgorithmGeosVelAtp** (p. 131), and **bratl::CBratAlgorithmGeosVelGrid** (p. 24).

6.14.3.6 `virtual string bratl::CBratAlgorithmBase::GetName ()` [pure virtual]

Gets the name of the algorithm

Implemented in **bratl::CBratAlgorithmGeosVelGridV** (p. 24), **bratl::CBratAlgorithmGeosVelGridU** (p. 24), **bratl::CBratAlgoFilterGaussian1D** (p. 102), **bratl::CBratAlgoFilterGaussian2D** (p. 104), **bratl::CBratAlgoFilterLanczos1D** (p. 105), **bratl::CBratAlgoFilterLanczos2D** (p. 107), **bratl::CBratAlgoFilterLoess1D**

(p. 110), **bratl::CBratAlgoFilterLoess2D** (p. 113), **bratl::CBratAlgoFilterMedian1D** (p. 116), **bratl::CBratAlgoFilterMedian2D** (p. 119), and **bratl::CBratAlgorithmGeosVelAtp** (p. 131).

6.14.3.7 `virtual uint32_t bratl::CBratAlgorithmBase::GetNumInputParam () [pure virtual]`

Gets the number of input parameters to pass to the 'Run' function

Implemented in **bratl::CBratAlgoFilterLoess1D** (p. 110), **bratl::CBratAlgoFilterLoess2D** (p. 113), **bratl::CBratAlgoFilterMedian1D** (p. 116), **bratl::CBratAlgoFilterMedian2D** (p. 119), **bratl::CBratAlgorithmGeosVelAtp** (p. 131), and **bratl::CBratAlgorithmGeosVelGrid** (p. 24).

6.14.3.8 `virtual string bratl::CBratAlgorithmBase::GetOutputUnit () [pure virtual]`

Gets the unit of an output value returned by the 'Run' function.

Implemented in **bratl::CBratAlgoFilterLoess1D** (p. 110), **bratl::CBratAlgoFilterLoess2D** (p. 113), **bratl::CBratAlgoFilterMedian2D** (p. 119), **bratl::CBratAlgoFilterMedian1D** (p. 116), **bratl::CBratAlgorithmGeosVelAtp** (p. 131), and **bratl::CBratAlgorithmGeosVelGrid** (p. 24).

6.14.3.9 `CBratAlgorithmBase & bratl::CBratAlgorithmBase::operator= (const CBratAlgorithmBase & o)`

Overloads operator '='

6.14.3.10 `virtual double bratl::CBratAlgorithmBase::Run (CVectorBratAlgorithmParam & args) [pure virtual]`

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

Returns

the result of the execution

Implemented in **bratl::CBratAlgoFilterLoess1D** (p. 111), **bratl::CBratAlgoFilterLoess2D** (p. 114), **bratl::CBratAlgoFilterMedian2D** (p. 120), **bratl::CBratAlgoFilterMedian1D** (p. 117), **bratl::CBratAlgorithmGeosVelAtp** (p. 131), **bratl::CBratAlgorithmGeosVelGrid** (p. 24), **bratl::CBratAlgoFilterGaussian1D** (p. 102), **bratl::CBratAlgoFilterGaussian2D** (p. 104), **bratl::CBratAlgoFilterLanczos1D** (p. 105), and **bratl::CBratAlgoFilterLanczos2D** (p. 107).

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

- BratAlgorithmBase.h
- BratAlgorithmBase.cpp

6.15 bratl::CBratAlgorithmGeosVel Class Reference

```
#include <BratAlgorithmGeosVel.h>
```


- virtual void **SetNextValues** ()
- virtual void **SetPreviousValues** (bool fromProduct)

Protected Attributes

- double **m_beta**
- double **m_coriolis**
- double **m_degreeToRadianMultiplier**
- double **m_earthRadius**
- bool **m_equatorTransition**
- bool **m_equatorTransitionIsNext**
- double **m_gravity**
- double **m_lat**
- **CDoubleArray** * **m_latArray**
- double **m_latNext**
- double **m_latPrev**
- double **m_lon**
- **CDoubleArray** * **m_lonArray**
- double **m_lonNext**
- double **m_lonPrev**
- double **m_omega**
- double **m_p2**
- double **m_velocity**

Static Protected Attributes

- static const string **m_LAT_PARAM_NAME** = "%{lat}"
- static const string **m_LON_PARAM_NAME** = "%{lon}"

Additional Inherited Members

6.15.1 Detailed Description

Algorithm base class.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 bratl::CBratAlgorithmGeosVel::CBratAlgorithmGeosVel ()

Default constructor

6.15.2.2 bratl::CBratAlgorithmGeosVel::CBratAlgorithmGeosVel (const CBratAlgorithmGeosVel & copy)

Copy constructor

6.15.2.3 bratl::CBratAlgorithmGeosVel::~CBratAlgorithmGeosVel () [virtual]

Destructor

6.15.3 Member Function Documentation

6.15.3.1 void bratl::CBratAlgorithmGeosVel::Dump (ostream & fOut = cerr) [virtual]

Dump function

Reimplemented from **bratl::CBratAlgorithmBase** (p. 123).

Reimplemented in **bratl::CBratAlgorithmGeosVelGridV** (p. 23), **bratl::CBratAlgorithmGeosVelGridU** (p. 23), **bratl::CBratAlgorithmGeosVelGrid** (p. 23), and **bratl::CBratAlgorithmGeosVelAtp** (p. 130).

References bratl::CBratAlgorithmBase::Dump().

Referenced by bratl::CBratAlgorithmGeosVelAtp::Dump(), and bratl::CBratAlgorithmGeosVelGrid::Dump().

6.15.3.2 CBratAlgorithmGeosVel & bratl::CBratAlgorithmGeosVel::operator= (const CBratAlgorithmGeosVel & copy)

Overloads operator '='

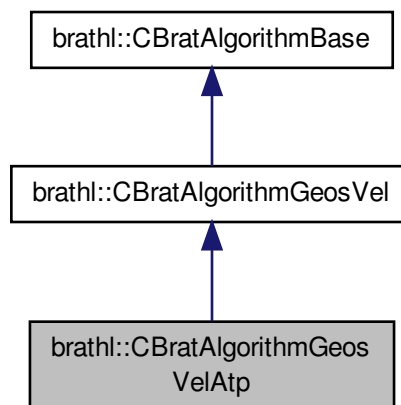
The documentation for this class was generated from the following files:

- BratAlgorithmGeosVel.h
- BratAlgorithmGeosVel.cpp

6.16 bratl::CBratAlgorithmGeosVelAtp Class Reference

```
#include <BratAlgorithmGeosVelAtp.h>
```

Inheritance diagram for bratl::CBratAlgorithmGeosVelAtp:



[illegible]

- **CBratAlgorithmGeosVelAtp** ()
- **CBratAlgorithmGeosVelAtp** (const **CBratAlgorithmGeosVelAtp** ©)
- virtual void **CheckInputParams** (CVectorBratAlgorithmParam &args)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual string **GetDescription** ()
- virtual string **GetInputParamDesc** (uint32_t indexParam)
- virtual
CBratAlgorithmParam::bratAlgoParamTypeVal **GetInputParamFormat** (uint32_t indexParam)
- virtual string **GetInputParamUnit** (uint32_t indexParam)
- virtual string **GetName** ()
- virtual uint32_t **GetNumInputParam** ()
- virtual string **GetOutputUnit** ()
- virtual string **GetParamName** (uint32_t indexParam)
- double **GetTrackDirection** ()
- **CBratAlgorithmGeosVelAtp** & **operator=** (const **CBratAlgorithmGeosVelAtp** ©)
- virtual double **Run** (CVectorBratAlgorithmParam &args)
- virtual void **SetParamValues** (CVectorBratAlgorithmParam &args)
- virtual ~**CBratAlgorithmGeosVelAtp** ()

- double **ComputeVelocity** ()
- double **ComputeVelocityEquator** ()
- double **ComputeVelocityOutsideEquator** ()
- void **Init** ()
- void **Set** (const **CBratAlgorithmGeosVelAtp** ©)
- void **SetBeginOfFile** ()
- void **SetEndOfFile** ()
- void **SetEquatorTransition** ()
- void **SetGap** ()
- virtual void **SetNextValues** ()
- virtual void **SetPreviousValues** (bool fromProduct)

- double **m_gap**
- double **m_varValue**
- double **m_varValueNext**
- double **m_varValuePrev**

Static Protected Attributes

- static const uint32_t **m_INPUT_PARAMS** = 3
- static const uint32_t **m_LAT_PARAM_INDEX** = 0
- static const uint32_t **m_LON_PARAM_INDEX** = 1
- static const uint32_t **m_VAR_PARAM_INDEX** = 2

Additional Inherited Members

6.16.1 Detailed Description

Algorithm base class.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 bratl::CBratAlgorithmGeosVelAtp::CBratAlgorithmGeosVelAtp ()

Default constructor

6.16.2.2 bratl::CBratAlgorithmGeosVelAtp::CBratAlgorithmGeosVelAtp (const CBratAlgorithmGeosVelAtp & copy)

Copy constructor

6.16.2.3 virtual bratl::CBratAlgorithmGeosVelAtp::~CBratAlgorithmGeosVelAtp () [inline],[virtual]

Destructor

6.16.3 Member Function Documentation

6.16.3.1 void bratl::CBratAlgorithmGeosVelAtp::Dump (ostream & fOut = cerr) [virtual]

Dump function

Reimplemented from **bratl::CBratAlgorithmGeosVel** (p. 128).

References **bratl::CBratAlgorithmGeosVel::Dump()**.

6.16.3.2 virtual string bratl::CBratAlgorithmGeosVelAtp::GetDescription () [inline],[virtual]

Gets the description of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.16.3.3 virtual string bratl::CBratAlgorithmGeosVelAtp::GetInputParamDesc (uint32_t indexParam) [inline],[virtual]

Gets the description of an input parameter.

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References **bratl::CTools::Format()**.

6.16.3.4 `virtual CBratAlgorithmParam::bratAlgoParamTypeVal bratl::CBratAlgorithmGeosVelAtp::GetInputParamFormat (uint32_t indexParam) [inline],[virtual]`

Gets the format of an input parameter : CBratAlgorithmParam::T_DOUBLE for double CBratAlgorithmParam::T_FLOAT for float CBratAlgorithmParam::T_INT for integer CBratAlgorithmParam::T_LONG for long integer CBratAlgorithmParam::T_STRING for string CBratAlgorithmParam::T_CHAR for a character

Parameters

<i>indexParam</i>	[in] : parameter index. First parameter index is 0, last one is 'number of parameters - 1'.
-------------------	---

Implements **bratl::CBratAlgorithmBase** (p. 124).

References bratl::CTools::Format().

6.16.3.5 `virtual string bratl::CBratAlgorithmGeosVelAtp::GetInputParamUnit (uint32_t indexParam) [inline],[virtual]`

Gets the unit of an input parameter :

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **bratl::CBratAlgorithmBase** (p. 124).

References bratl::CTools::Format().

6.16.3.6 `virtual string bratl::CBratAlgorithmGeosVelAtp::GetName () [inline],[virtual]`

Gets the name of the algorithm

Implements **bratl::CBratAlgorithmBase** (p. 124).

6.16.3.7 `virtual uint32_t bratl::CBratAlgorithmGeosVelAtp::GetNumInputParam () [inline],[virtual]`

Gets the number of input parameters to pass to the 'Run' function

Implements **bratl::CBratAlgorithmBase** (p. 125).

6.16.3.8 `virtual string bratl::CBratAlgorithmGeosVelAtp::GetOutputUnit () [inline],[virtual]`

Gets the unit of an output value returned by the 'Run' function.

Parameters

<i>indexParam</i>	[in] : parameter index.
-------------------	-------------------------

Implements **bratl::CBratAlgorithmBase** (p. 125).

6.16.3.9 `CBratAlgorithmGeosVelAtp & bratl::CBratAlgorithmGeosVelAtp::operator= (const CBratAlgorithmGeosVelAtp & copy)`

Overloads operator '='

6.16.3.10 `double bratl::CBratAlgorithmGeosVelAtp::Run (CVectorBratAlgorithmParam & args) [virtual]`

Runs the algorithm

Parameters

<i>fmt</i>	[in] : a string that indicates the format of each value of input parameters (number, string) : d for integer l for long integer f for double s for string
<i>args</i>	[in] : the values of input parameters i(a C/C++ va_list).

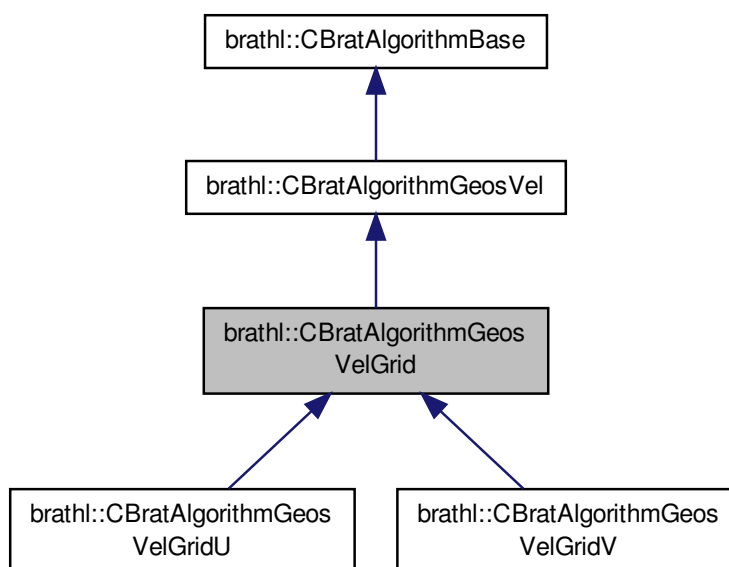
the result of the execution

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

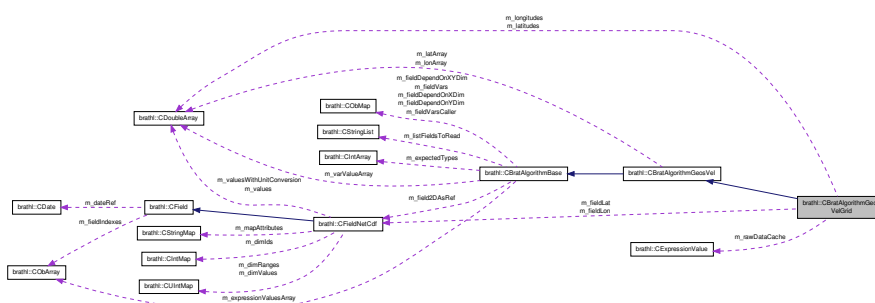
- BratAlgorithmGeosVelAtp.h
- BratAlgorithmGeosVelAtp.cpp

```
#include <BratAlgorithmGeosVelGrid.h>
```

Inheritance diagram for brathl::CBratAlgorithmGeosVelGrid:



Collaboration diagram for brathl::CBratAlgorithmGeosVelGrid:



Public Member Functions

- **CBratAlgorithmGeosVelGrid** ()
- **CBratAlgorithmGeosVelGrid** (const **CBratAlgorithmGeosVelGrid** ©)
- virtual void **CheckInputParams** (CVectorBratAlgorithmParam &args)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual string **GetInputParamDesc** (uint32_t indexParam)
- virtual
CBratAlgorithmParam::bratAlgoParamTypeVal **GetInputParamFormat** (uint32_t indexParam)
- virtual string **GetInputParamUnit** (uint32_t indexParam)
- virtual uint32_t **GetNumInputParam** ()
- virtual string **GetOutputUnit** ()
- virtual double **GetParamDefaultValue** (uint32_t indexParam)
- virtual string **GetParamName** (uint32_t indexParam)
- **CBratAlgorithmGeosVelGrid** & **operator=** (const **CBratAlgorithmGeosVelGrid** ©)
- virtual double **Run** (CVectorBratAlgorithmParam &args)
- virtual void **SetParamValues** (CVectorBratAlgorithmParam &args)
- virtual ~**CBratAlgorithmGeosVelGrid** ()

Protected Member Functions

- void **CheckEquatorLimit** ()
- void **CheckLatLonExpression** (uint32_t index)
- void **CheckProduct** ()
- void **CheckVarExpression** (uint32_t index)
- double **ComputeMean** ()
- double **ComputeSingle** ()
- virtual double **ComputeVelocity** ()=0
- virtual void **DeleteFieldNetCdf** ()
- virtual void **DeleteProduct** ()
- uint32_t **GetLatDimRange** (CFieldNetCdf *field)
- int32_t **GetLatitudeIndex** (double value)
- void **GetLatitudes** ()
- uint32_t **GetLonDimRange** (CFieldNetCdf *field)
- int32_t **GetLongitudeIndex** (double value)
- void **GetLongitudes** ()
- void **GetVarCacheExpressionValue** (int32_t minIndexLat, int32_t maxIndexLat, int32_t minIndexLon, int32_t maxIndexLon)
- double **GetVarExpressionValue** (int32_t indexLat, int32_t indexLon)
- double **GetVarExpressionValueCache** (int32_t indexLat, int32_t indexLon)
- void **Init** ()
- virtual void **OpenProductFile** ()
- bool **PrepareComputeVelocity** ()
- virtual void **PrepareDataReading2D** (int32_t minIndexLat, int32_t maxIndexLat, int32_t minIndexLon, int32_t maxIndexLon)
- virtual void **PrepareDataReading2D** (int32_t indexLat, int32_t indexLon)
- virtual void **PrepareDataValues2DComplexExpression** (CExpressionValue &exprValue)
- virtual void **PrepareDataValues2DComplexExpressionWithAlgo** (CExpressionValue &exprValue)
- virtual void **PrepareDataValues2DOneField** (CExpressionValue &exprValue)
- void **Set** (const **CBratAlgorithmGeosVelGrid** ©)
- void **SetBeginOfFile** ()
- void **SetEndOfFile** ()

Protected Attributes

- bool **m_allLongitudes**
- double **m_equatorLimit**
- **CFieldNetCdf * m_fieldLat**
- **CFieldNetCdf * m_fieldLon**
- int32_t **m_indexLat**
- int32_t **m_indexLon**
- **CDoubleArray m_latitudes**
- **CDoubleArray m_longitudes**
- double **m_ionMax**
- double **m_ionMin**
- **CExpressionValue m_rawDataCache**
- int32_t **m_varDimLatIndex**
- int32_t **m_varDimLonIndex**
- double **m_varValue**
- double **m_varValueE**
- double **m_varValueN**
- double **m_varValueS**
- double **m_varValueW**

Static Protected Attributes

- static const uint32_t **m_EQUATOR_LAT_LIMIT_INDEX** = 3
- static const uint32_t **m_INPUT_PARAMS** = 4
- static const uint32_t **m_LAT_PARAM_INDEX** = 0
- static const uint32_t **m_LON_PARAM_INDEX** = 1
- static const uint32_t **m_VAR_PARAM_INDEX** = 2

Additional Inherited Members

6.17.1 Detailed Description

Algorithm base class.

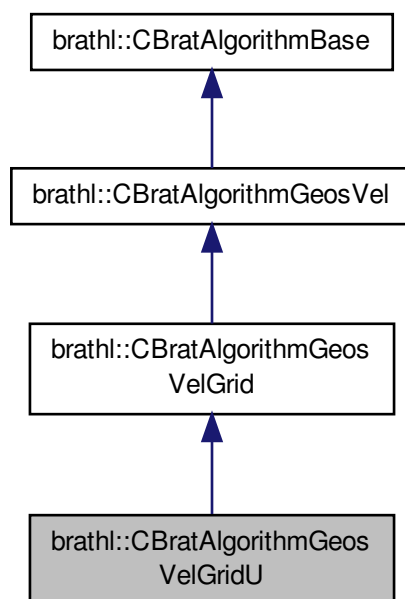
The documentation for this class was generated from the following files:

- BratAlgorithmGeosVelGrid.h
- BratAlgorithmGeosVelGrid.cpp

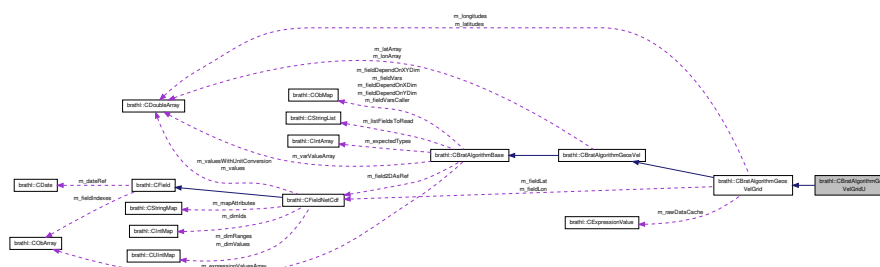
6.18 bratl::CBratAlgorithmGeosVelGridU Class Reference

```
#include <BratAlgorithmGeosVelGrid.h>
```

Inheritance diagram for brathl::CBratAlgorithmGeosVelGridU:



Collaboration diagram for brathl::CBratAlgorithmGeosVelGridU:



Public Member Functions

- **CBratAlgorithmGeosVelGridU** ()
- **CBratAlgorithmGeosVelGridU** (const **CBratAlgorithmGeosVelGridU** ©)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual string **GetDescription** ()
- virtual string **GetName** ()
- virtual ~**CBratAlgorithmGeosVelGridU** ()

Protected Member Functions

- double **ComputeVelocity** ()
- void **Init** ()

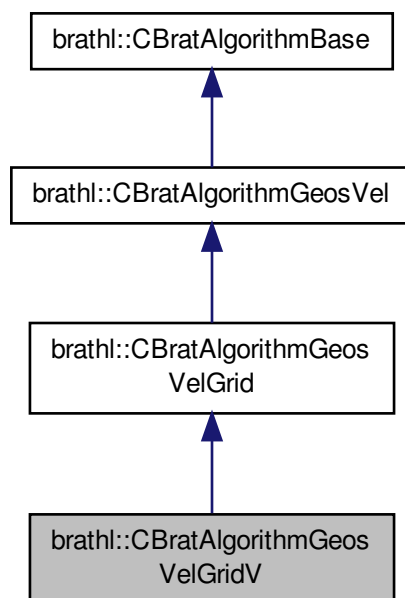
6.18.1 Detailed Description

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

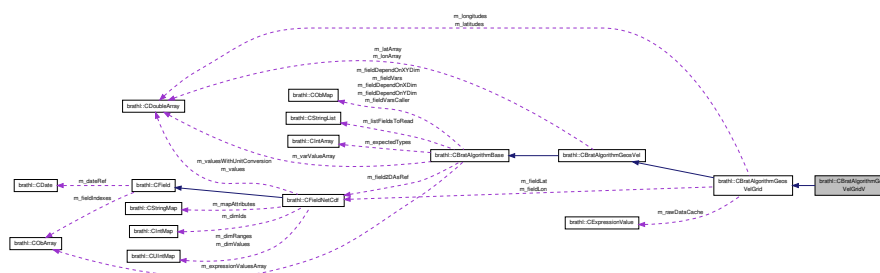
- BratAlgorithmGeosVelGrid.h
- BratAlgorithmGeosVelGrid.cpp

```
#include <BratAlgorithmGeosVelGrid.h>
```

Inheritance diagram for brathl::CBratAlgorithmGeosVelGridV:



Collaboration diagram for brathl::CBratAlgorithmGeosVelGridV:



Public Member Functions

- **CBratAlgorithmGeosVelGridV** ()
- **CBratAlgorithmGeosVelGridV** (const **CBratAlgorithmGeosVelGridV** ©)
- virtual void **Dump** (ostream &fOut=cerr)
- virtual string **GetDescription** ()
- virtual string **GetName** ()
- virtual ~**CBratAlgorithmGeosVelGridV** ()

Protected Member Functions

- double **ComputeVelocity** ()
- void **Init** ()

Additional Inherited Members

6.19.1 Detailed Description

Algorithm base class.

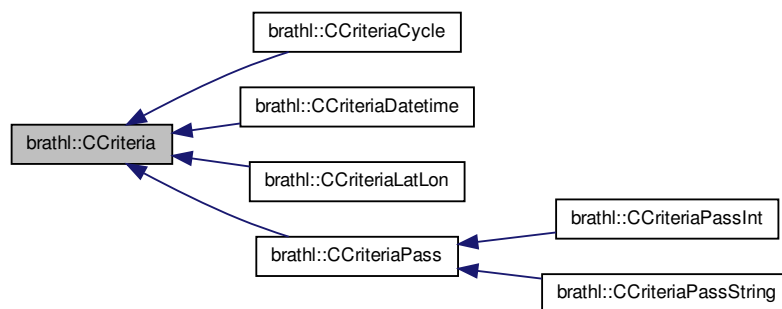
The documentation for this class was generated from the following files:

- BratAlgorithmGeosVelGrid.h
- BratAlgorithmGeosVelGrid.cpp

6.20 bratl::CCriteria Class Reference

```
#include <Criteria.h>
```

Inheritance diagram for bratl::CCriteria:



Public Types

- enum **CriteriaKind** {
 UNKNOWN, **LATLON**, **DATETIME**, **PASS**,
 CYCLE }

Public Member Functions

- **CCriteria** ()
*Empty **CCriteria** (p. 137) ctor.*
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual string **GetAsText** (const string &delimiter)=0
- int32_t **GetKey** ()
- virtual bool **IsDefaultValue** ()=0
- virtual void **SetDefaultValue** ()=0
- virtual ~**CCriteria** ()
Destructor.

Static Public Member Functions

- static void **Adjust** (CIntArray &array)
- static **CCriteria** * **GetCriteria** (CBratObject *ob, bool withExcept=true)

Protected Attributes

- int32_t **m_key**

6.20.1 Detailed Description

Criteria management class.

Version

1.0

6.20.2 Member Enumeration Documentation

6.20.2.1 enum brathl::CCriteria::CriteriaKind

Kind of criteria enumeration.

Enumerator:

UNKNOWN not set
LATLON geographical latitude/longitude area
DATETIME date/time
PASS Pass
CYCLE Cycle

6.20.3 Member Function Documentation

6.20.3.1 virtual bool brathl::CCriteria::IsDefaultValue () [pure virtual]

Tests whether value have been initialized or not

Returns

true if not initialized

Implemented in **brathl::CCriteriaPassInt** (p. 80), **brathl::CCriteriaLatLon** (p. 157), **brathl::CCriteriaDatetime** (p. 148), **brathl::CCriteriaCycle** (p. 142), **brathl::CCriteriaPassString** (p. 79), and **brathl::CCriteriaPass** (p. 79).

6.20.3.2 virtual void brathl::CCriteria::SetDefaultValue () [pure virtual]

Sets internal value to the default value (uninitialized)

Implemented in **brathl::CCriteriaPassInt** (p. 81), **brathl::CCriteriaLatLon** (p. 158), **brathl::CCriteriaDatetime** (p. 149), **brathl::CCriteriaCycle** (p. 143), **brathl::CCriteriaPassString** (p. 81), and **brathl::CCriteriaPass** (p. 81).

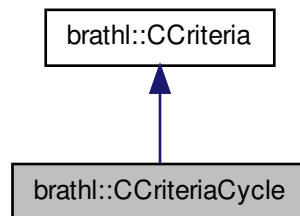
The documentation for this class was generated from the following files:

- Criteria.h
- Criteria.cpp

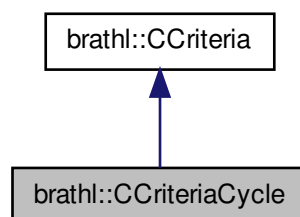
6.21 brathl::CCriteriaCycle Class Reference

```
#include <CriteriaCycle.h>
```

Inheritance diagram for brathl::CCriteriaCycle:



Collaboration diagram for brathl::CCriteriaCycle:



Public Member Functions

- **CCriteriaCycle** ()
Empty **CCriteriaCycle** (p. 139) ctor.
- **CCriteriaCycle** (CCriteriaCycle &c)
- **CCriteriaCycle** (CCriteriaCycle *c)
- **CCriteriaCycle** (int32_t from, int32_t to)

- **CCriteriaCycle** (const string &from, const string &to)
 - **CCriteriaCycle** (const CStringArray &array)
 - virtual void **Dump** (ostream &fOut=cerr)
- Dump fonction.*
- string **GetAsText** (const string &delimiter=CCriteriaCycle::m_delimiter)
 - int32_t **GetFrom** ()
 - int32_t **GetTo** ()
 - bool **Intersect** (CStringArray &array, CStringArray &intersect)
 - bool **Intersect** (CStringArray &array, **CIntArray** &intersect)
 - bool **Intersect** (**CIntArray** &array, CStringArray &intersect)
 - bool **Intersect** (**CIntArray** &array, **CIntArray** &intersect)
 - bool **Intersect** (int32_t from, int32_t to, CStringArray &intersect)
 - bool **Intersect** (int32_t from, int32_t to, **CIntArray** &intersect)
 - bool **Intersect** (const string &from, const string &to, **CIntArray** &intersect)
 - bool **Intersect** (double otherFrom, double otherTo, **CIntArray** &intersect)
 - bool **Intersect** (const string &from, const string &to, CStringArray &intersect)
 - bool **IsDefaultValue** ()
 - const **CCriteriaCycle** & **operator=** (**CCriteriaCycle** &c)
 - void **Set** (**CCriteriaCycle** &c)
 - void **Set** (int32_t from, int32_t to)
 - void **Set** (const string &from, const string &to)
 - void **Set** (const CStringArray &array)
 - void **SetDefaultValue** ()
 - void **SetFrom** (int32_t from)
 - void **SetFrom** (const string &from)
 - void **SetFromText** (const string &values, const string &delimiter=CCriteriaCycle::m_delimiter)
 - void **SetTo** (int32_t to)
 - void **SetTo** (const string &to)
 - virtual ~**CCriteriaCycle** ()

Destructor.

Static Public Member Functions

- static **CCriteriaCycle** * **GetCriteria** (CBratObject *ob, bool withExcept=true)

Static Public Attributes

- static const string **m_delimiter** = " "

Protected Member Functions

- void **Adjust** ()
- void **Init** ()

Protected Attributes

- int32_t **m_from**
- int32_t **m_to**

Additional Inherited Members

6.21.1 Detailed Description

Pass number (from/to) Criteria management class.

Version

1.0

6.21.2 Constructor & Destructor Documentation

6.21.2.1 brathl::CCriteriaCycle::CCriteriaCycle (int32_t *from*, int32_t *to*)

Constructor.

Parameters

<i>from</i>	start pass
<i>to</i>	end pass

6.21.2.2 brathl::CCriteriaCycle::CCriteriaCycle (const string & *from*, const string & *to*)

Constructor.

Parameters

<i>from</i>	start pass
<i>to</i>	end pass

6.21.2.3 brathl::CCriteriaCycle::CCriteriaCycle (const CStringArray & *array*)

Constructor from a array that contains start pass as string, end pass as string

Parameters

<i>array</i>	start and end dates
--------------	---------------------

6.21.3 Member Function Documentation

6.21.3.1 bool brathl::CCriteriaCycle::Intersect (CStringArray & *array*, CStringArray & *intersect*)

Create the intersection of this date period with the given one

Parameters

<i>array</i>	that contains start pass as string, end pass as string
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

6.21.3.2 bool brathl::CCriteriaCycle::Intersect (CStringArray & *array*, CIntArray & *intersect*)

Create the intersection of this date period with the given one

Parameters

<i>array</i>	that contains start pass as string, end pass as string
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

6.21.3.3 bool brathl::CCriteriaCycle::Intersect (CIntArray & array, CStringArray & intersect)

Create the intersection of this date period with the given one

Parameters

<i>array</i>	that contains start pass as string, end pass as string
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

6.21.3.4 bool brathl::CCriteriaCycle::Intersect (CIntArray & array, CIntArray & intersect)

Create the intersection of this date period with the given one

Parameters

<i>array</i>	that contains start pass as string, end pass as string
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

6.21.3.5 bool brathl::CCriteriaCycle::IsDefaultValue () [virtual]

Tests whether the pass have been initialized or not

Returns

true if not initialized

Implements **brathl::CCriteria** (p. 138).

6.21.3.6 void brathl::CCriteriaCycle::Set (int32_t from, int32_t to)

Sets date period from start and end pass

Parameters

<i>from</i>	start pass
<i>to</i>	end pass

6.21.3.7 void brathl::CCriteriaCycle::Set (const string & from, const string & to)

Sets date period from start and end pass

Parameters

<i>from</i>	start pass
<i>to</i>	end pass

References brathl::CTools::StrToInt().

6.21.3.8 void brathl::CCriteriaCycle::Set (const CStringArray & array)

Sets a date period from a array that contains start pass as string, end pass as string

Parameters

<i>array</i>	start and end dates
--------------	---------------------

6.21.3.9 void brathl::CCriteriaCycle::SetDefaultValue () [virtual]

Sets internal value to the default value (uninitialized)

Implements **brathl::CCriteria** (p. 139).

6.21.3.10 void brathl::CCriteriaCycle::SetFrom (int32_t from)

Sets start pass

Parameters

<i>to</i>	start pass
-----------	------------

6.21.3.11 void brathl::CCriteriaCycle::SetFrom (const string & from)

Sets start pass

Parameters

<i>to</i>	start pass
-----------	------------

References brathl::CTools::StrToInt().

6.21.3.12 void brathl::CCriteriaCycle::SetTo (int32_t to)

Sets end pass

Parameters

<i>to</i>	end pass
-----------	----------

6.21.3.13 void brathl::CCriteriaCycle::SetTo (const string & to)

Sets end pass

Parameters

<i>to</i>	end pass
-----------	----------

References brathl::CTools::StrToInt().

6.21.4 Member Data Documentation

6.21.4.1 `int32_t brathl::CCriteriaCycle::m_from` [protected]

start pass

6.21.4.2 `int32_t brathl::CCriteriaCycle::m_to` [protected]

end pass

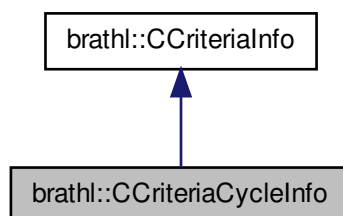
The documentation for this class was generated from the following files:

- CriteriaCycle.h
- CriteriaCycle.cpp

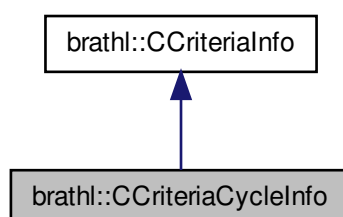
6.22 brathl::CCriteriaCycleInfo Class Reference

```
#include <CriteriaInfo.h>
```

Inheritance diagram for brathl::CCriteriaCycleInfo:



Collaboration diagram for brathl::CCriteriaCycleInfo:



Public Member Functions

- **CCriteriaCycleInfo** ()
Empty CCriteriaCycleInfo (p. 144) ctor.
- virtual void **Dump** (ostream &fOut=cerr)

Dump function.

- CFieldInfo * **GetEndCycleField** ()
- const string & **GetEndCycleFieldName** ()
- virtual void **GetFieldsInfo** (CObMap *fieldsInfo)
- CFieldInfo * **GetStartCycleField** ()
- const string **GetStartCycleFieldName** ()
- void **SetEndCycleField** (const string &value)
- void **SetEndCycleField** (CFieldInfo &value)
- void **SetStartCycleField** (const string &value)
- void **SetStartCycleField** (CFieldInfo &value)
- virtual ~**CCriteriaCycleInfo** ()

Destructor.

Static Public Member Functions

- static **CCriteriaCycleInfo** * **GetCriteriaInfo** (CBratObject *ob, bool withExcept=true)

Protected Attributes

- CFieldInfo **m_endCycleField**
- CFieldInfo **m_startCycleField**

6.22.1 Detailed Description

Cycle criteria information management class.

Version

1.0

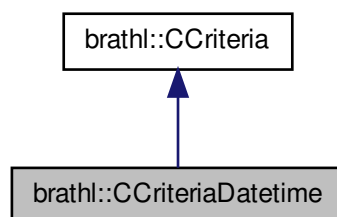
The documentation for this class was generated from the following files:

- CriteriaInfo.h
- CriteriaInfo.cpp

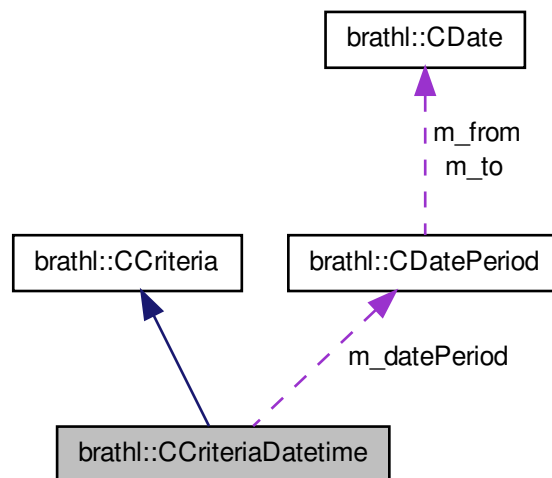
6.23 brathl::CCriteriaDatetime Class Reference

```
#include <CriteriaDatetime.h>
```

Inheritance diagram for brathl::CCriteriaDatetime:



Collaboration diagram for brathl::CCriteriaDatetime:



Public Member Functions

- **CCriteriaDatetime** ()
Empty CCriteriaDatetime (p. 145) ctor.
- **CCriteriaDatetime** (CCriteriaDatetime &c)
- **CCriteriaDatetime** (CCriteriaDatetime *c)
- **CCriteriaDatetime** (CDatePeriod &datePeriod)
- **CCriteriaDatetime** (CDate &from, CDate &to)
- **CCriteriaDatetime** (const string &from, const string &to)
- **CCriteriaDatetime** (double from, double to)
- **CCriteriaDatetime** (const CStringArray &array)
- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- string **GetAsText** (const string &delimiter=CDatePeriod::m_delimiter)
- CDatePeriod * **GetDatePeriod** ()
- CDate * **GetFrom** ()
- string **GetFromAsText** ()
- CDate * **GetTo** ()
- string **GetToAsText** ()
- bool **Intersect** (CDatePeriod &datePeriod, CDatePeriod &intersect)
- bool **Intersect** (double otherFrom, double otherTo, CDatePeriod &intersect)
- bool **Intersect** (double otherFrom, double otherTo)
- bool **IsDefaultValue** ()
- const CCriteriaDatetime & **operator=** (CCriteriaDatetime &c)
- void **Set** (CDatePeriod &datePeriod)
- void **Set** (CDate &from, CDate &to)
- void **Set** (const string &from, const string &to)
- void **Set** (double from, double to)
- void **Set** (const CStringArray &array)
- void **Set** (CCriteriaDatetime &c)

- void **SetDefaultValue** ()
- void **SetFrom** (CDate &from)
- void **SetFrom** (const string &strDate)
- void **SetFromText** (const string &values, const string &delimiter=CDatePeriod::m_delimiter)
- void **SetTo** (CDate &to)
- void **SetTo** (const string &strDate)
- virtual **~CCriteriaDatetime** ()

Destructor.

Static Public Member Functions

- static **CCriteriaDatetime** * **GetCriteria** (CBratObject *ob, bool withExcept=true)

Protected Member Functions

- void **Init** ()

Protected Attributes

- **CDatePeriod** m_datePeriod

Additional Inherited Members

6.23.1 Detailed Description

Datetime Criteria management class.

Version

1.0

6.23.2 Constructor & Destructor Documentation

6.23.2.1 bratl::CCriteriaDatetime::CCriteriaDatetime (CDatePeriod & datePeriod)

Constructor.

Parameters

<i>datePeriod</i>	period to set
-------------------	---------------

6.23.2.2 bratl::CCriteriaDatetime::CCriteriaDatetime (CDate & from, CDate & to)

Constructor.

Parameters

<i>from</i>	start date
<i>to</i>	end date

6.23.2.3 bratl::CCriteriaDatetime::CCriteriaDatetime (const string & from, const string & to)

Constructor.

Parameters

<i>from</i>	start date
<i>to</i>	end date

6.23.2.4 brathl::CCriteriaDatetime::CCriteriaDatetime (double *from*, double *to*)

Constructor.

Parameters

<i>from</i>	start date (number of seconds since 1950-01-01)
<i>to</i>	end date (number of seconds since 1950-01-01)

6.23.2.5 brathl::CCriteriaDatetime::CCriteriaDatetime (const CStringArray & *array*)

Constructor from a array that contains start date as string, end date as string

Parameters

<i>array</i>	start and end dates
--------------	---------------------

6.23.3 Member Function Documentation

6.23.3.1 bool brathl::CCriteriaDatetime::Intersect (CDatePeriod & *datePeriod*, CDatePeriod & *intersect*)

Create the intersection of this date period with the given one

Parameters

<i>datePeriod</i>	intersect with this
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

6.23.3.2 bool brathl::CCriteriaDatetime::Intersect (double *otherFrom*, double *otherTo*, CDatePeriod & *intersect*)

Create the intersection of this date period with the given one

Parameters

<i>otherFrom</i>	start date intersect with this
<i>otherTo</i>	end date intersect with this
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

6.23.3.3 bool brathl::CCriteriaDatetime::IsDefaultValue () [virtual]

Tests whether date period have been initialized or not

Returns

true if not initialized

Implements **brathl::CCriteria** (p. 138).

6.23.3.4 void brathl::CCriteriaDatetime::Set (CDatePeriod & *datePeriod*)

Sets date period from another one

Parameters

<i>datePeriod</i>	period to set
-------------------	---------------

6.23.3.5 void brathl::CCriteriaDatetime::Set (CDate & *from*, CDate & *to*)

Sets date period from start and end date

Parameters

<i>from</i>	start date
<i>to</i>	end date

6.23.3.6 void brathl::CCriteriaDatetime::Set (const string & *from*, const string & *to*)

Sets date period from start and end date

Parameters

<i>from</i>	start date
<i>to</i>	end date

6.23.3.7 void brathl::CCriteriaDatetime::Set (const CStringArray & *array*)

Sets a date period from a array that contains start date as string, end date as string

Parameters

<i>array</i>	start and end dates
--------------	---------------------

6.23.3.8 void brathl::CCriteriaDatetime::SetDefaultValue () [virtual]

Sets internal value to the default value (uninitialized)

Implements **brathl::CCriteria** (p. 139).

6.23.3.9 void brathl::CCriteriaDatetime::SetFrom (CDate & *from*)

Sets start date

Parameters

<i>to</i>	start date
-----------	------------

6.23.3.10 void brathl::CCriteriaDatetime::SetFrom (const string & *strDate*)

Sets start date

Parameters

<i>to</i>	start date
-----------	------------

6.23.3.11 void brathl::CCriteriaDatetime::SetTo (CDate & *to*)

Sets end date

Parameters

<i>to</i>	end date
-----------	----------

6.23.3.12 void brathl::CCriteriaDatetime::SetTo (const string & *strDate*)

Sets end date

Parameters

<i>to</i>	end date
-----------	----------

6.23.4 Member Data Documentation

6.23.4.1 CDatePeriod brathl::CCriteriaDatetime::m_datePeriod [protected]

Date period

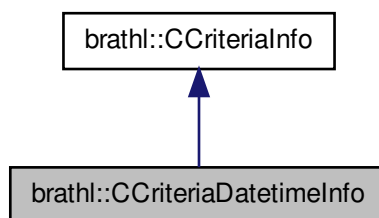
The documentation for this class was generated from the following files:

- CriteriaDatetime.h
- CriteriaDatetime.cpp

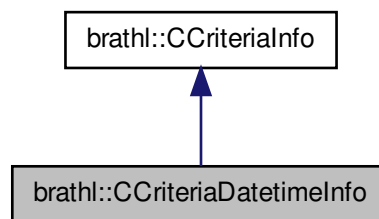
6.24 brathl::CCriteriaDatetimeInfo Class Reference

```
#include <CriteriaInfo.h>
```

Inheritance diagram for brathl::CCriteriaDatetimeInfo:



Collaboration diagram for brathl::CCriteriaDatetimedInfo:



Public Member Functions

- **CCriteriaDatetimedInfo ()**
Empty CCriteriaDatetimedInfo (p. 150) ctor.
- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- CFieldInfo * **GetEndDateField** ()
- const string & **GetEndDateFieldName** ()
- virtual void **GetFieldsInfo** (CObMap *fieldsInfo)
- **brathl_refDate** **GetRefDate** ()
- CFieldInfo * **GetStartDateField** ()
- const string & **GetStartDateFieldName** ()
- void **SetEndDateField** (const string &value)
- void **SetEndDateField** (CFieldInfo &value)
- void **SetRefDate** (brathl_refDate value)
- void **SetStartDateField** (const string &value)
- void **SetStartDateField** (CFieldInfo &value)
- virtual ~**CCriteriaDatetimedInfo** ()
Destructor.

Static Public Member Functions

- static **CCriteriaDatetimedInfo** * **GetCriterialInfo** (CBratObject *ob, bool withExcept=true)

Protected Attributes

- CFieldInfo **m_endDateField**
- **brathl_refDate** **m_refDate**
- CFieldInfo **m_startDateField**

6.24.1 Detailed Description

Date/Time criteria information management class.

Version

1.0

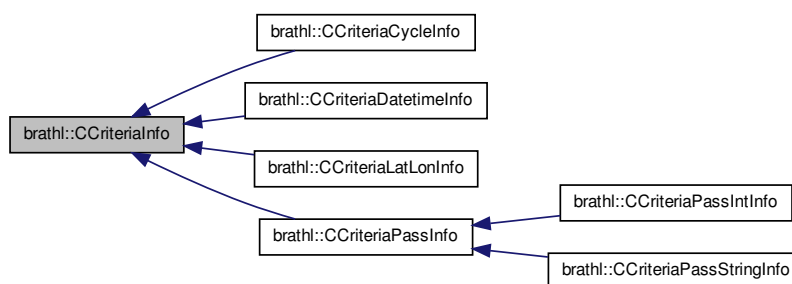
The documentation for this class was generated from the following files:

- CriterialInfo.h
- CriterialInfo.cpp

6.25 bratl::CCriterialInfo Class Reference

```
#include <CriteriaInfo.h>
```

Inheritance diagram for bratl::CCriterialInfo:



Public Member Functions

- **CCriterialInfo** ()
*Empty **CCriterialInfo** (p. 152) ctor.*
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- string **GetDataRecord** ()
- virtual void **GetFieldNames** (CStringList &fieldNames)
- virtual void **GetFieldNames** (CStringArray &fieldNames)
- virtual void **GetFields** (CRecordDataMap &listRecord)
- virtual void **GetFieldsInfo** (CObMap *fieldsInfo)=0
- int32_t **GetKey** ()
- void **SetDataRecord** (const string &value)
- virtual ~**CCriterialInfo** ()
Destructor.

Static Public Member Functions

- static **CCriterialInfo** * **GetCriterialInfo** (CBratObject *ob, bool withExcept=true)

Protected Attributes

- string **m_dataRecord**
- int32_t **m_key**

6.25.1 Detailed Description

Base class for criteria information.

Version

1.0

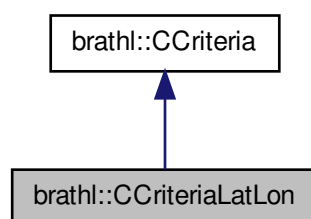
The documentation for this class was generated from the following files:

- CriteriaInfo.h
- CriteriaInfo.cpp

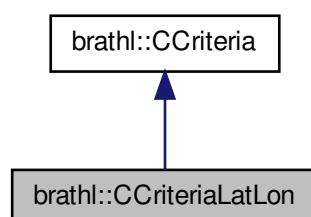
6.26 brathl::CCriteriaLatLon Class Reference

```
#include <CriteriaLatLon.h>
```

Inheritance diagram for brathl::CCriteriaLatLon:



Collaboration diagram for brathl::CCriteriaLatLon:



Public Member Functions

- **CCriteriaLatLon ()**
Empty CCriteriaLatLon (p. 153) ctor.
- **CCriteriaLatLon (CCriteriaLatLon &c)**

- **CCriteriaLatLon** (**CCriteriaLatLon** *c)
- **CCriteriaLatLon** (CLatLonRect &latLonRect)
- **CCriteriaLatLon** (CLatLonPoint &p1, double deltaLat, double deltaLon)
- **CCriteriaLatLon** (CLatLonPoint &latLonLow, CLatLonPoint &latLonHigh)
- **CCriteriaLatLon** (double latLow, double lonLow, double latHigh, double lonHigh)
- **CCriteriaLatLon** (const string &latLow, const string &lonLow, const string &latHigh, const string &lonHigh)
- **CCriteriaLatLon** (const CStringArray &array)
- virtual void **Dump** (ostream &fOut=cerr)

Dump fonction.

- virtual string **GetAsText** (const string &delimiter=CLatLonRect::m_delimiter)
- CLatLonRect * **GetLatLonRect** ()
- double **GetLowerLeftLat** ()
- double **GetLowerLeftLon** ()
- double **GetLowerRightLat** ()
- double **GetLowerRightLon** ()
- double **GetUpperLeftLat** ()
- double **GetUpperLeftLon** ()
- double **GetUpperRightLat** ()
- double **GetUpperRightLon** ()
- bool **Intersect** (CLatLonRect &clip, CLatLonRect &intersect)
- bool **IsDefaultValue** ()
- const **CCriteriaLatLon** & **operator=** (**CCriteriaLatLon** &c)
- void **Set** (CLatLonRect &latLonRect)
- void **Set** (CLatLonPoint &p1, double deltaLat, double deltaLon)
- void **Set** (CLatLonPoint &latLonLow, CLatLonPoint &latLonHigh)
- void **Set** (double latLow, double lonLow, double latHigh, double lonHigh)
- void **Set** (const string &latLow, const string &lonLow, const string &latHigh, const string &lonHigh)
- void **Set** (const string &latLonRect, const string &delimiter=CLatLonRect::m_delimiter)
- void **Set** (**CCriteriaLatLon** &c)
- void **SetDefaultValue** ()
- virtual ~**CCriteriaLatLon** ()

Destructor.

Static Public Member Functions

- static **CCriteriaLatLon** * **GetCriteria** (CBratObject *ob, bool withExcept=true)
- static double **GetMinOrMaxLon** (double lon1, double lon2, bool wantMin)

Protected Member Functions

- void **Init** ()

Protected Attributes

- CLatLonRect **m_latLonRect**

Additional Inherited Members

6.26.1 Detailed Description

Latitude/Longitude Criteria management class.

Version

1.0

6.26.2 Constructor & Destructor Documentation

6.26.2.1 bratl::CCriteriaLatLon::CCriteriaLatLon (CLatLonRect & *latLonRect*)

Constructor.

Parameters

<i>latLonRect</i>	lat/lon bounding box
-------------------	----------------------

6.26.2.2 bratl::CCriteriaLatLon::CCriteriaLatLon (CLatLonPoint & *p1*, double *deltaLat*, double *deltaLon*)

Construct a lat/lon bounding box from a point, and a delta lat, lon. This disambiguates which way the box wraps around the globe.

Parameters

<i>p1</i>	one corner of the box
<i>deltaLat</i>	delta lat from p1. (may be positive or negative)
<i>deltaLon</i>	delta lon from p1. (may be positive or negative)

6.26.2.3 bratl::CCriteriaLatLon::CCriteriaLatLon (CLatLonPoint & *latLonLow*, CLatLonPoint & *latLonHigh*)

Constructor.

Parameters

<i>latLonLow</i>	lat/lon low point
<i>latLonHigh</i>	lat/lon high point

6.26.2.4 bratl::CCriteriaLatLon::CCriteriaLatLon (double *latLow*, double *lonLow*, double *latHigh*, double *lonHigh*)

Constructor.

Parameters

<i>latLow</i>	latitude low
<i>lonLow</i>	longitude low
<i>latHigh</i>	latitude high
<i>lonHigh</i>	longitude high

6.26.2.5 bratl::CCriteriaLatLon::CCriteriaLatLon (const string & *latLow*, const string & *lonLow*, const string & *latHigh*, const string & *lonHigh*)

Constructor.

Parameters

<i>latLow</i>	latitude low
<i>lonLow</i>	longitude low
<i>latHigh</i>	latitude high
<i>lonHigh</i>	longitude high

6.26.2.6 bratl::CCriteriaLatLon::CCriteriaLatLon (const CStringArray & *array*)

Constructor from a list that contains low latitude value, low longitude value, high latitude value, high longitude value.

Parameters

<i>array</i>	to be converted
--------------	-----------------

6.26.2.7 bratl::CCriteriaLatLon::~~CCriteriaLatLon () [virtual]

Destructor.

Getter of the property `latLonRect`.

Returns

Returns the `latLonRect`.

6.26.3 Member Function Documentation

6.26.3.1 double bratl::CCriteriaLatLon::GetLowerLeftLat () [inline]

Returns

lower left latitude of the lat/lon box, `Double.MAX_VALUE` if not set.

6.26.3.2 double bratl::CCriteriaLatLon::GetLowerLeftLon () [inline]

Returns

lower left longitude of the lat/lon box, `Double.MAX_VALUE` if not set.

6.26.3.3 double bratl::CCriteriaLatLon::GetLowerRightLat () [inline]

Returns

lower right latitude of the lat/lon box, `Double.MAX_VALUE` if not set.

6.26.3.4 double bratl::CCriteriaLatLon::GetLowerRightLon () [inline]

Returns

lower right longitude of the lat/lon box, `Double.MAX_VALUE` if not set.

6.26.3.5 double bratl::CCriteriaLatLon::GetMinOrMaxLon (double lon1, double lon2, bool wantMin) [static]

Gets the min. or max. of two longitudes.

Parameters

<i>lon1</i>	first longitude
<i>lon2</i>	second longitude
<i>wantMin</i>	true: returns min., false: returns max.

Returns

min. lon or max. lon, depends on `wantMin`.

References `bratl::CTools::Max()`, and `bratl::CTools::Min()`.

6.26.3.6 double bratl::CCriteriaLatLon::GetUpperLeftLat () [inline]

Returns

upper left latitude of the lat/lon box, Double.MAX_VALUE if not set.

6.26.3.7 double brathl::CCriteriaLatLon::GetUpperLeftLon () [inline]

Returns

upper left longitude of the lat/lon box, Double.MAX_VALUE if not set.

6.26.3.8 double brathl::CCriteriaLatLon::GetUpperRightLat () [inline]

Returns

upper right latitude of the lat/lon box, Double.MAX_VALUE if not set.

6.26.3.9 double brathl::CCriteriaLatLon::GetUpperRightLon () [inline]

Returns

upper right longitude of the lat/lon box, Double.MAX_VALUE if not set.

6.26.3.10 bool brathl::CCriteriaLatLon::Intersect (CLatLonRect & *clip*, CLatLonRect & *intersect*)

Create the intersection of this LatLon Criteria with the given one

Parameters

<i>clip</i>	intersect with this
<i>intersection</i>	

Returns

true, or false if there is no intersection

6.26.3.11 bool brathl::CCriteriaLatLon::IsDefaultValue () [virtual]

Tests whether date period have been initialized or not

Returns

true if not initialized

Implements **brathl::CCriteria** (p. 138).

6.26.3.12 void brathl::CCriteriaLatLon::Set (CLatLonRect & *latLonRect*)

Setter of the property `latLonRect`;

Parameters

<i>latLonRect</i>	The latLonRect to set.
-------------------	------------------------

6.26.3.13 void brathl::CCriteriaLatLon::Set (CLatLonPoint & *p1*, double *deltaLat*, double *deltaLon*)

Set a lat/lon bounding box from a point, and a delta lat, lon. This disambiguates which way the box wraps around the globe.

Parameters

<i>p1</i>	one corner of the box
<i>deltaLat</i>	delta lat from p1. (may be positive or negative)
<i>deltaLon</i>	delta lon from p1. (may be positive or negative)

6.26.3.14 void brathl::CCriteriaLatLon::Set (CLatLonPoint & *latLonLow*, CLatLonPoint & *latLonHigh*)

Setter of the property `atLonRect`;

Parameters

<i>latLonLow</i>	lat/lon low point
<i>latLonHigh</i>	lat/lon high point .property name="latLonRect"

6.26.3.15 void brathl::CCriteriaLatLon::Set (double *latLow*, double *lonLow*, double *latHigh*, double *lonHigh*)

Setter of the property `atLonRect`;

Parameters

<i>latLow</i>	latitude low
<i>lonLow</i>	longitude low
<i>latHigh</i>	latitude high
<i>lonHigh</i>	longitude high

6.26.3.16 void brathl::CCriteriaLatLon::Set (const string & *latLow*, const string & *lonLow*, const string & *latHigh*, const string & *lonHigh*)

Setter of the property `atLonRect`;

Parameters

<i>latLow</i>	latitude low
<i>lonLow</i>	longitude low
<i>latHigh</i>	latitude high
<i>lonHigh</i>	longitude high

6.26.3.17 void brathl::CCriteriaLatLon::Set (const string & *latLonRect*, const string & *delimiter* = CLatLonRect::m_delimiter)

Setter of the property `atLonRect`;

Parameters

<i>latLonRect</i>	latitude low, longitude low, latitude high, longitude high
-------------------	--

6.26.3.18 void brathl::CCriteriaLatLon::SetDefaultValue () [virtual]

Sets internal value to the default value (uninitialized)

Implements **brathl::CCriteria** (p. 139).

6.26.4 Member Data Documentation

6.26.4.1 CLatLonRect bratl::CCriteriaLatLon::m_latLonRect [protected]

Bounding box for latitude/longitude points. This is a rectangle in lat/lon coordinates. Note that LatLonPoint always has lon in the range +/-180. *

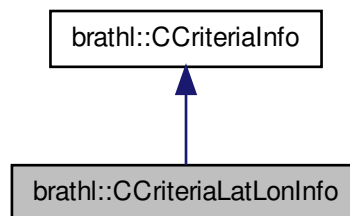
The documentation for this class was generated from the following files:

- CCriteriaLatLon.h
- CCriteriaLatLon.cpp

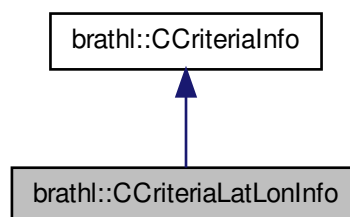
6.27 bratl::CCriteriaLatLonInfo Class Reference

```
#include <CriteriaInfo.h>
```

Inheritance diagram for bratl::CCriteriaLatLonInfo:



Collaboration diagram for bratl::CCriteriaLatLonInfo:



Public Member Functions

- **CCriteriaLatLonInfo** ()
Empty CCriteriaLatLonInfo (p. 159) ctor.
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- CFieldInfo * **GetEndLatField** ()
- const string & **GetEndLatFieldName** ()

- CFieldInfo * **GetEndLonField** ()
- const string & **GetEndLonFieldName** ()
- virtual void **GetFieldsInfo** (CObMap *fieldsInfo)
- CFieldInfo * **GetStartLatField** ()
- const string & **GetStartLatFieldName** ()
- CFieldInfo * **GetStartLonField** ()
- const string & **GetStartLonFieldName** ()
- void **SetEndLatField** (const string &value)
- void **SetEndLatField** (CFieldInfo &value)
- void **SetEndLonField** (const string &value)
- void **SetEndLonField** (CFieldInfo &value)
- void **SetStartLatField** (const string &value)
- void **SetStartLatField** (CFieldInfo &value)
- void **SetStartLonField** (const string &value)
- void **SetStartLonField** (CFieldInfo &value)
- virtual ~**CCriteriaLatLonInfo** ()

Destructor.

Static Public Member Functions

- static **CCriteriaLatLonInfo** * **GetCriteriaInfo** (CBratObject *ob, bool withExcept=true)

Protected Attributes

- CFieldInfo **m_endLatField**
- CFieldInfo **m_endLonField**
- CFieldInfo **m_startLatField**
- CFieldInfo **m_startLonField**

6.27.1 Detailed Description

Lat/Lon criteria information management class.

Version

1.0

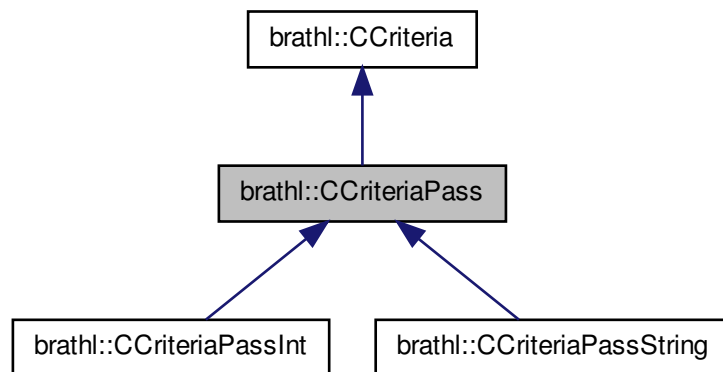
The documentation for this class was generated from the following files:

- CriteriaInfo.h
- CriteriaInfo.cpp

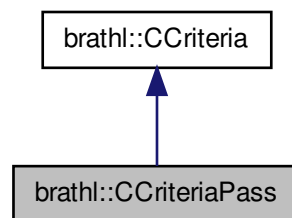
6.28 brathl::CCriteriaPass Class Reference

```
#include <CriteriaPass.h>
```

Inheritance diagram for bratl::CCriteriaPass:



Collaboration diagram for bratl::CCriteriaPass:



Public Member Functions

- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual bool **IsDefaultValue** ()=0
- virtual void **SetDefaultValue** ()=0
- virtual **~CCriteriaPass** ()
Destructor.

Static Public Member Functions

- static **CCriteriaPass** * **GetCriteria** (CBratObject *ob, bool withExcept=true)

Protected Member Functions

- **CCriteriaPass** ()

Empty **CCriteriaPass** (p. 160) ctor.

- void **Init** ()

Additional Inherited Members

6.28.1 Detailed Description

Pass number Criteria management class.

Version

1.0

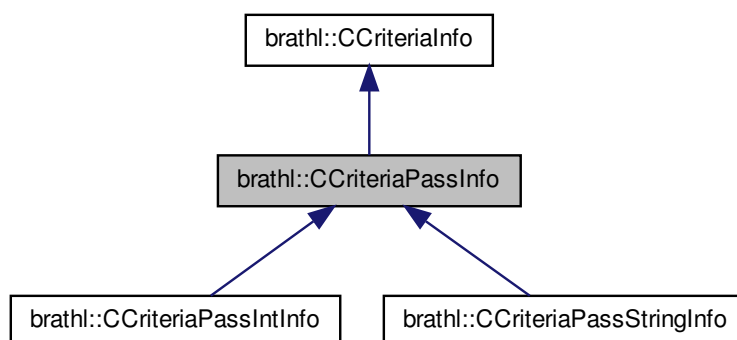
The documentation for this class was generated from the following files:

- CriteriaPass.h
- CriteriaPass.cpp

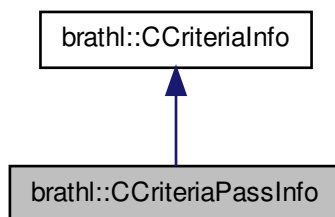
6.29 bratl::CCriteriaPassInfo Class Reference

```
#include <CriteriaInfo.h>
```

Inheritance diagram for bratl::CCriteriaPassInfo:



Collaboration diagram for brathl::CCriteriaPassInfo:



Public Member Functions

- **CCriteriaPassInfo** ()
Empty CCriteriaPassInfo (p. 162) ctor.
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- CFieldInfo * **GetEndPassField** ()
- const string & **GetEndPassFieldName** ()
- virtual void **GetFieldsInfo** (CObMap *fieldsInfo)
- CFieldInfo * **GetStartPassField** ()
- const string & **GetStartPassFieldName** ()
- void **SetEndPassField** (const string &value)
- void **SetEndPassField** (CFieldInfo &value)
- void **SetStartPassField** (const string &value)
- void **SetStartPassField** (CFieldInfo &value)
- virtual ~**CCriteriaPassInfo** ()
Destructor.

Static Public Member Functions

- static **CCriteriaPassInfo** * **GetCriteriaInfo** (CBratObject *ob, bool withExcept=true)

Protected Attributes

- CFieldInfo **m_endPassField**
- CFieldInfo **m_startPassField**

6.29.1 Detailed Description

Pass criteria information management class.

Version

1.0

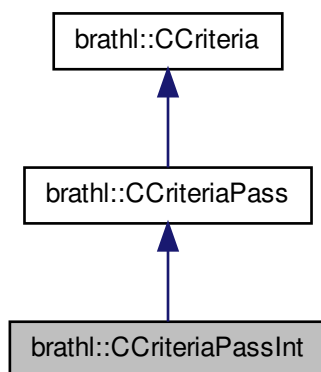
The documentation for this class was generated from the following files:

- CriteriaInfo.h
- CriteriaInfo.cpp

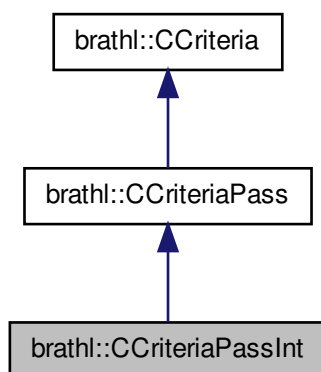
6.30 bratl::CCriteriaPassInt Class Reference

```
#include <CriteriaPass.h>
```

Inheritance diagram for bratl::CCriteriaPassInt:



Collaboration diagram for bratl::CCriteriaPassInt:



Public Member Functions

- **CCriteriaPassInt** ()
Empty CCriteriaPassInt (p. 164) ctor.
- **CCriteriaPassInt** (CCriteriaPassInt &c)
- **CCriteriaPassInt** (CCriteriaPassInt *c)
- **CCriteriaPassInt** (int32_t from, int32_t to)
- **CCriteriaPassInt** (const string &from, const string &to)
- **CCriteriaPassInt** (const CStringArray &array)

- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- string **GetAsText** (const string &delimiter=CCriteriaPassInt::m_delimiter)
- int32_t **GetFrom** ()
- int32_t **GetTo** ()
- bool **Intersect** (CStringArray &array, CStringArray &intersect)
- bool **Intersect** (CStringArray &array, **CIntArray** &intersect)
- bool **Intersect** (**CIntArray** &array, CStringArray &intersect)
- bool **Intersect** (**CIntArray** &array, **CIntArray** &intersect)
- bool **Intersect** (int32_t from, int32_t to, CStringArray &intersect)
- bool **Intersect** (int32_t from, int32_t to, **CIntArray** &intersect)
- bool **Intersect** (double otherFrom, double otherTo, **CIntArray** &intersect)
- bool **Intersect** (const string &from, const string &to, **CIntArray** &intersect)
- bool **Intersect** (const string &from, const string &to, CStringArray &intersect)
- bool **IsDefaultValue** ()
- const **CCriteriaPassInt** & **operator=** (**CCriteriaPassInt** &c)
- void **Set** (**CCriteriaPassInt** &c)
- void **Set** (int32_t from, int32_t to)
- void **Set** (const string &from, const string &to)
- void **Set** (const CStringArray &array)
- void **SetDefaultValue** ()
- void **SetFrom** (int32_t from)
- void **SetFrom** (const string &from)
- void **SetFromText** (const string &values, const string &delimiter=CCriteriaPassInt::m_delimiter)
- void **SetTo** (int32_t to)
- void **SetTo** (const string &to)
- virtual ~**CCriteriaPassInt** ()
Destructor.

Static Public Member Functions

- static **CCriteriaPassInt** * **GetCriteria** (CBratObject *ob, bool withExcept=true)

Static Public Attributes

- static const string **m_delimiter** = " "

Protected Member Functions

- void **Adjust** ()
- void **Init** ()

Protected Attributes

- int32_t **m_from**
- int32_t **m_to**

Additional Inherited Members

6.30.1 Detailed Description

Pass number (from/to) Criteria management class.

Version

1.0

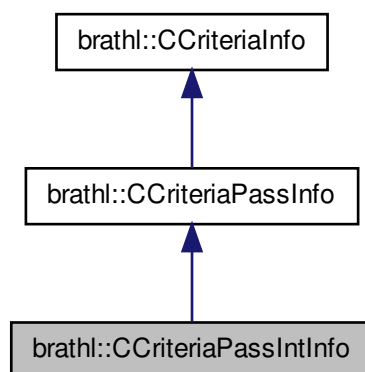
The documentation for this class was generated from the following files:

- CriteriaPass.h
- CriteriaPass.cpp

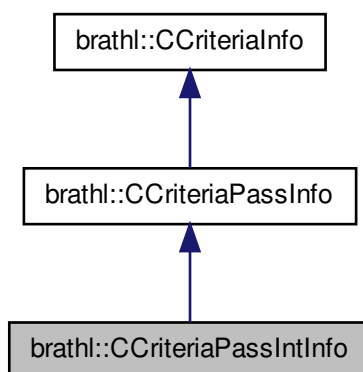
6.31 brathl::CCriteriaPassIntInfo Class Reference

```
#include <CriteriaInfo.h>
```

Inheritance diagram for brathl::CCriteriaPassIntInfo:



Collaboration diagram for bratl::CCriteriaPassIntInfo:



Public Member Functions

- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.

Static Public Member Functions

- static **CCriteriaPassIntInfo** * **GetCriterialInfo** (CBratObject *ob, bool withExcept=true)

Additional Inherited Members

6.31.1 Detailed Description

Integer Pass criteria information management class.

Version

1.0

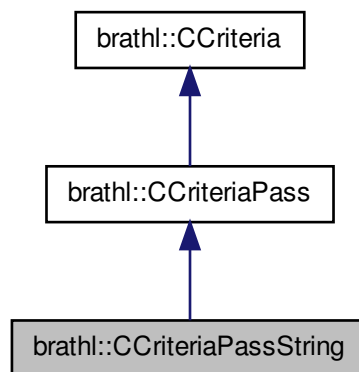
The documentation for this class was generated from the following files:

- CriterialInfo.h
- CriterialInfo.cpp

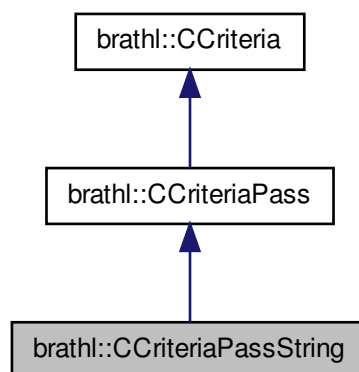
6.32 bratl::CCriteriaPassString Class Reference

```
#include <CriteriaPass.h>
```

Inheritance diagram for bratl::CCriteriaPassString:



Collaboration diagram for bratl::CCriteriaPassString:



Public Member Functions

- **CCriteriaPassString** ()
Empty CCriteriaPassString (p. 167) ctor.
- **CCriteriaPassString** (CCriteriaPassString &c)
- **CCriteriaPassString** (CCriteriaPassString *c)
- **CCriteriaPassString** (const string &passes, const string &delimiter=CCriteriaPassString::m_delimiter)
- **CCriteriaPassString** (const CStringArray &array)
- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- string **GetAsText** (const string &delimiter=CCriteriaPassString::m_delimiter)
- CStringArray * **GetPasses** ()

- bool **Intersect** (const string &passes, CStringArray &intersect)
- bool **Intersect** (CStringArray &passes, CStringArray &intersect)
- bool **IsDefaultValue** ()
- const **CCriteriaPassString** & **operator=** (**CCriteriaPassString** &c)
- void **Set** (const string &passes, const string &delimiter=CCriteriaPassString::m_delimiter)
- void **Set** (const CStringArray &array)
- void **Set** (**CCriteriaPassString** &c)
- void **SetDefaultValue** ()
- virtual ~**CCriteriaPassString** ()

Destructor.

Static Public Member Functions

- static **CCriteriaPassString** * **GetCriteria** (CBratObject *ob, bool withExcept=true)

Static Public Attributes

- static const string **m_delimiter** = ","

Protected Member Functions

- void **Init** ()

Static Protected Member Functions

- static void **ExtractPass** (const string &passes, CStringArray &arrayPass, const string &delimiter=CCriteriaPassString::m_delimiter)
- static void **ExtractPass** (const CStringArray &array, CStringArray &arrayPass)

Protected Attributes

- CStringArray **m_passes**

Additional Inherited Members

6.32.1 Detailed Description

Pass number (as string) Criteria management class.

Version

1.0

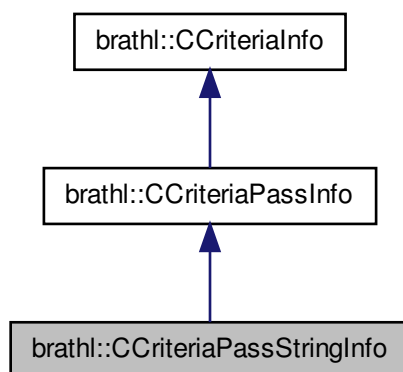
The documentation for this class was generated from the following files:

- CriteriaPass.h
- CriteriaPass.cpp

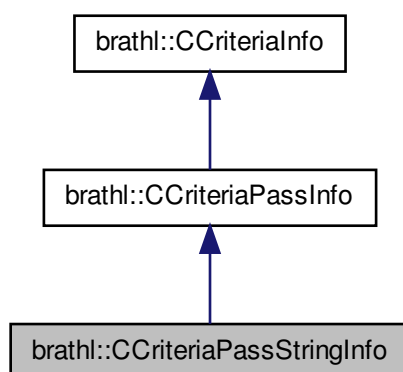
6.33 bratl::CCriteriaPassStringInfo Class Reference

```
#include <CriteriaInfo.h>
```

Inheritance diagram for bratl::CCriteriaPassStringInfo:



Collaboration diagram for bratl::CCriteriaPassStringInfo:



Public Member Functions

- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.

Static Public Member Functions

- static **CCriteriaPassStringInfo** * **GetCriteriaInfo** (CBratObject *ob, bool withExcept=true)

Additional Inherited Members

6.33.1 Detailed Description

String Pass criteria information management class.

Version

1.0

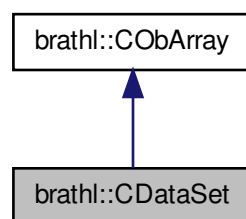
The documentation for this class was generated from the following files:

- CriteriaInfo.h
- CriteriaInfo.cpp

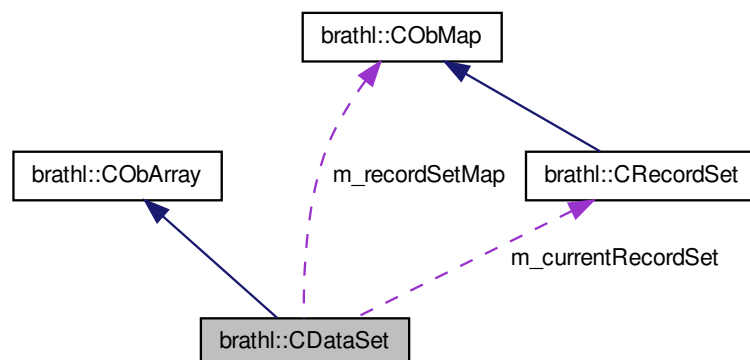
6.34 brathl::CDataSet Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CDataSet:



Collaboration diagram for brathl::CDataSet:



Public Member Functions

- **CRecordSet * Back** (bool withExcept=true)
- **CDataSet** (const string &name="", bool bDelete=true)
Ctor.
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual bool **Erase** (CRecordSet *recordSet)
- bool **EraseCurrentRecordSet** ()
- void **EraseFieldSet** (const string &fieldSetKey)
- **CRecordSet * FindRecord** (const string &recordSetName)
- **CRecordSet * GetCurrentRecordSet** ()
- **CFieldSet * GetFieldSet** (const string &fieldSetKey)
- **CFieldSetArrayDbI * GetFieldSetAsArrayDbI** (const string &fieldSetKey)
- **CFieldSetDbI * GetFieldSetAsDbI** (const string &fieldSetKey)
- double **GetFieldSetAsDbIValue** (const string &fieldSetKey)
- **CFieldSetString * GetFieldSetAsString** (const string &fieldSetKey)
- string **GetFieldSetAsStringValue** (const string &fieldSetKey)
- **CRecordSet * GetFirstRecordSet** ()
- const string & **GetName** ()
- **CRecord * GetRecord** (const string &recordSetName)
- **CRecord * GetRecord** (CRecordSet *recordSet)
- **CRecordSet * GetRecordSet** (CDataSet::iterator itDataSet)
- **CRecordSet * GetRecordSet** (int32_t index)
- **CObMap * GetRecordSetMap** ()
- void **InsertDataset** (CDataSet *dataSet, bool setAsCurrent=true)
- void **InsertFieldSet** (const string &fieldSetKey, CFieldSet *fieldSet)
- **CRecordSet * InsertRecord** (const string &recordSetName, bool setAsCurrent=true)
- virtual void **RemoveAll** ()
- void **SetCurrentRecordSet** (int32_t index)
- void **SetCurrentRecordSet** (CDataSet::iterator itDataSet)
- void **SetCurrentRecordSet** (const string &recordSetName)
- void **SetCurrentRecordSet** (CRecordSet *recordSet)
- void **SetName** (const string &name)
- virtual ~**CDataSet** ()
Dtor.

Protected Attributes

- **CRecordSet * m_currentRecordSet**
- string **m_name**
- **CObMap m_recordSetMap**

6.34.1 Detailed Description

a set of recordset management classes.

Version

1.0

6.34.2 Member Function Documentation

6.34.2.1 void brathl::CDataSet::Dump (ostream & fOut = cerr) [virtual]

Dump fonction.

Copy a new **CDataSet** (p. 171) to the object

Referenced by EraseFieldSet(), and InsertFieldSet().

6.34.2.2 void brathl::CDataSet::EraseFieldSet (const string & fieldSetKey)

remove a fieldset object (identify by its name) from the current recordset

Parameters

<i>fieldSetKey</i>	[in] : fieldset key
--------------------	---------------------

References BRATHL_LOGIC_ERROR, Dump(), brathl::CObMap::Erase(), and brathl::CTools::Format().

6.34.2.3 CFieldSet * brathl::CDataSet::GetFieldSet (const string & fieldSetKey)

Gets the fieldset object (identify by its name) of the current recordset

Parameters

<i>fieldSetKey</i>	[in] : fieldset key to be searched
--------------------	------------------------------------

Returns

a pointer to the fieldset object if found, otherwise NULL

6.34.2.4 void brathl::CDataSet::InsertFieldSet (const string & fieldSetKey, CFieldSet * fieldSet)

Inserts a fieldset object (identify by its name) into the current recordset

Parameters

<i>fieldSetKey</i>	[in] : fieldset key
<i>fieldSet</i>	[in] : fieldset object to be inserted

References BRATHL_LOGIC_ERROR, Dump(), brathl::CTools::Format(), and brathl::CObMap::Insert().

6.34.2.5 void brathl::CDataSet::RemoveAll () [virtual]

Remove all elements and clear the list

Reimplemented from **brathl::CObArray** (p. 63).

References brathl::CObMap::RemoveAll().

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

- Field.h
- Field.cpp

6.35 brathl::CDate Class Reference

```
#include <Date.h>
```

Public Member Functions

- **int32_t Add** (const **CDate** &d)
- **int32_t AddDays** (uint32_t days)
- **string AsString** (const string &format="", bool withMuSecond=false)
- **CDate** ()
Constructs a date with a 1950/01/01 value.
- **CDate** (const char *strDate)
- **CDate** (const **CDate** &date)
*Constructs a date from another **CDate** (p. 173) object.*
- **CDate** (const uint32_t year, const uint32_t month=1, const uint32_t day=1, const uint32_t hour=0, const uint32_t minute=0, const uint32_t second=0, const uint32_t muSecond=0)
Constructs a date from year, month, day, hour, minute, second, microsecond.
- **CDate** (const uint32_t days, const uint32_t seconds, const uint32_t muSeconds, const **brathl_refDate** refDate=**REF19500101**)
Constructs a date from days, seconds, microseconds.
- **CDate** (const double days, const double seconds, const double muSeconds, const **brathl_refDate** refDate=**REF19500101**)
Constructs a date from days, seconds, microseconds.
- **CDate** (const double dateSeconds, **brathl_refDate** refDate=**REF19500101**)
- **CDate** (**brathl_refDate** refDate)
- **int32_t ConstructDate** (const **brathl_refDate** refDate)
- **int32_t Convert2DecimalJulian** (double &julian, const **brathl_refDate** refDate=**REF19500101**)
- **int32_t Convert2DMM** (int32_t &days, int32_t &milliSeconds, int32_t &muSeconds, const **brathl_refDate** refDate=**REF19500101**)
- **int32_t Convert2DMM** (double &days, double &milliSeconds, double &muSeconds, const **brathl_refDate** refDate=**REF19500101**)
- **int32_t Convert2DSM** (int32_t &days, int32_t &seconds, int32_t &muSeconds, const **brathl_refDate** refDate=**REF19500101**)
- **int32_t Convert2DSM** (double &days, double &seconds, double &muSeconds, const **brathl_refDate** refDate=**REF19500101**)
- **int32_t Convert2Second** (double &seconds, const **brathl_refDate** refDate=**REF19500101**)
- **int32_t Convert2SM** (int32_t &seconds, int32_t &muSeconds, const **brathl_refDate** refDate=**REF19500101**)
- **int32_t Convert2SM** (double &seconds, double &muSeconds, const **brathl_refDate** refDate=**REF19500101**)
- **int32_t Convert2YMDHMSM** (uint32_t &year, uint32_t &month, uint32_t &day, uint32_t &hour, uint32_t &minute, uint32_t &second, uint32_t &muSecond)
- **uint32_t DayOfYear** ()
- **virtual void Dump** (ostream &fOut=cerr)
Dump fonction.
- **uint32_t GetDay** ()
Gets the day of the date.
- **uint32_t GetHour** ()
Gets the hour of the date.
- **uint32_t GetMinute** ()
Gets the minutes of the date.
- **uint32_t GetMonth** ()
Gets the month of the date.
- **uint32_t GetMuSecond** ()
Gets the microseconds of the date.
- **uint32_t GetSecond** ()
Gets the seconds of the date.
- **uint32_t GetYear** ()
Gets the year of the date.

- uint32_t **HowManyLeapYear** (const uint32_t year)
- void **InitDateZero** ()
- bool **IsDefaultValue** ()
- bool **IsLeapYear** ()
- int32_t **LeapYearIndex** ()
- double **operator+** (CDate &d)
- double **operator-** (CDate &d)
- const CDate & **operator=** (const CDate &date)
- const CDate & **operator=** (const char *strDate)
- const CDate & **operator=** (double seconds)
- const CDate & **operator=** (const brathl_refDate refDate)
- int32_t **SetDate** (const char *strDate)
- int32_t **SetDate** (const brathl_DateYMDHMSM &date)
- int32_t **SetDate** (const brathl_DateDSM &date)
- int32_t **SetDate** (const uint32_t days, const uint32_t seconds, const uint32_t muSeconds, const brathl_refDate refDate=REF19500101)
- int32_t **SetDate** (const double days, const double seconds, const double muSeconds, const brathl_refDate refDate=REF19500101)
- int32_t **SetDate** (const brathl_DateSecond &date)
- int32_t **SetDate** (const brathl_DateJulian &date)
- int32_t **SetDate** (const uint32_t year, const uint32_t month=1, const uint32_t day=1, const uint32_t hour=0, const uint32_t minute=0, const uint32_t second=0, const uint32_t muSecond=0)
- int32_t **SetDate** (const double dateSeconds, brathl_refDate refDate=REF19500101)
- int32_t **SetDateJulian** (const double dateJulian, brathl_refDate refDate=REF19500101)
- int32_t **SetDateNow** ()
- void **SetDefaultValue** ()
- int32_t **SubtractDays** (uint32_t days)
- double **Value** ()
returns the date in a number of seconds since internal reference date, ie 1950)
- double **ValueJulian** ()
returns the date in a decimal julian day (since internal reference date, ie 1950)
- bool **operator<** (CDate &d)
- bool **operator<** (double d)
- bool **operator>** (CDate &d)
- bool **operator>** (double d)
- bool **operator==** (CDate &d)
- bool **operator==** (double d)
- bool **operator<=** (CDate &d)
- bool **operator<=** (double d)
- bool **operator>=** (CDate &d)
- bool **operator>=** (double d)
- bool **operator!=** (CDate &d)
- bool **operator!=** (double d)

Static Public Member Functions

- static int32_t **CheckDate** (const uint32_t year, const uint32_t month=1, const uint32_t day=1, const uint32_t hour=0, const uint32_t minute=0, const uint32_t second=0, const uint32_t muSecond=0)
- static int32_t **CheckDay** (uint32_t day, uint32_t month, uint32_t year)
- static int32_t **CheckHour** (uint32_t hour)
- static int32_t **CheckMinute** (uint32_t minute)
- static int32_t **CheckMonth** (uint32_t month)
- static int32_t **CheckMuSecond** (uint32_t muSecond)

- static int32_t **CheckSecond** (uint32_t second)
- static int32_t **CheckYear** (uint32_t year)
- static double **CvDate** (const char *strDate)
- static uint32_t **DayOfYear** (uint32_t year, uint32_t month, uint32_t day)
- static uint32_t **DayOfYear** (CDate &date)
- static int32_t **GetDaysInMonth** (const uint32_t month, const uint32_t year, uint32_t &nbDaysInMonth)
- static bool **IsCharDate** (const char *strDate)
- static bool **IsLeapYear** (const uint32_t year)
- static int32_t **LeapYearIndex** (const uint32_t year)

Static Public Attributes

- static const uint32_t **m_daysInMonth** [2][12]
- static const uint32_t **m_daysOfYear** [2][12]
- static const char * **m_DEFAULT_UNIT_SECOND** = "second"
- static const uint32_t **m_internalRefYear** = 1950
- static const double **m_minutesInDay** = 1440.0
- static const double **m_minutesInHour** = 60.0
- static const double **m_secInDay** = 86400.0
- static const double **m_secInHour** = 3600.0
- static const double **m_secInMinute** = 60.0

6.35.1 Detailed Description

Date management and conversion class.

This class allows calendar an date conversion.

Warning

Date before 1950/01/01 00:00:00:00 are not accepted

Version

1.0

6.35.2 Constructor & Destructor Documentation

6.35.2.1 brathl::CDate::CDate (const char * *strDate*)

Constructs a date from a string

Parameters

<i>strDate</i>	: Allowed format are : <ul style="list-style-type: none"> • YYYY-MM-DD HH:MM:SS.MS string • a julian string (format:positive 'Days Seconds Microseconds' or positive decimal julian day)
----------------	--

6.35.2.2 brathl::CDate::CDate (const double *dateSeconds*, brathl_refDate *refDate* = REF19500101)

Constructs a date value from a decimal number of seconds

Parameters

<i>dateSeconds</i>	[in]: decimal number of seconds
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

6.35.3 Member Function Documentation

6.35.3.1 `int32_t brathl::CDate::Add (const CDate & d)`

Adds a date to the date object

Parameters

<i>d</i>	[in]: a CDate (p. 173) object to add
----------	---

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**.

6.35.3.2 `int32_t brathl::CDate::AddDays (uint32_t days)`

Adds a number of day to the date object

Parameters

<i>days</i>	[in]: number of days to add (if < 0, a subtract operation is performed)
-------------	---

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, and `m_minutesInDay`.

6.35.3.3 `string brathl::CDate::AsString (const string & format = " ", bool withMuSecond = false)`

Formats a date as string.

Parameters

<i>Format</i>	[in] : String controlling how the date will be converted into string. This format string consists of zero or more conversion specifications and ordinary characters. A conversion specification consists of a " (percent) character and one or two terminating conversion characters that determine the conversion specification's behavior. All ordinary characters are copied unchanged into the result. Each conversion specification is replaced by appropriate characters as described in the following list. The appropriate characters are determined by the LC_TIME category of the program's locale. %% Same as %. a Locale's abbreviated weekday name. A Locale's full weekday name. b Locale's abbreviated month name. B Locale's full month name. c Locale's appropriate date and time representation. C Century number (the year divided by 100 and truncated to an integer as a decimal number [1,99]); single digits are preceded by 0; see standards(5). d Day of month [1,31]; single digits are preceded by 0. H Hour (24-hour clock) [0,23]; single digits are preceded by 0. I Hour (12-hour clock) [1,12]; single digits are preceded by 0. j Day number of year [1,366]; single digits are preceded by 0. m Month number [1,12]; single digits are preceded by 0. M Minute [00,59]; leading 0 is permitted but not required. p Locale's equivalent of either a.m. or p.m. S Seconds [00,61]; the range of values is [00,61] rather than [00,59] to allow for the occasional leap second and even more occasional double leap second. U Week number of year as a decimal number [00,53], with Sunday as the first day of week 1. w Weekday as a decimal number [0,6], with 0 representing Sunday. W Week number of year as a decimal number [00,53], with Monday as the first day of week 1. x Locale's appropriate date representation. X Locale's appropriate time representation. y Year within century [00,99]. Y Year, including the century (for example 1993). Z Time zone name or abbreviation, or no bytes if no time zone information exists. If the format is an empty string it is forced to be "%Y-%m-%d %H:%M:%S" (ISO 8601)
<i>withMuSecond</i>	[in] : add the microseconds of the date at the end of the string (format "%.06u")

Returns

Formatted string

References brathl::CTools::Format().

6.35.3.4 `int32_t brathl::CDate::CheckDate (const uint32_t year, const uint32_t month = 1, const uint32_t day = 1, const uint32_t hour = 0, const uint32_t minute = 0, const uint32_t second = 0, const uint32_t muSecond = 0) [static]`

Check if a date value (year, month, day, hour, minute, second, microsecond) is valid

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS.

6.35.3.5 `int32_t brathl::CDate::CheckDay (uint32_t day, uint32_t month, uint32_t year) [static]`

Checks if a day value is valid, according to a month an a year

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_ERROR_INVALID_DAY, and BRATHL_SUCCESS.

6.35.3.6 `int32_t brathl::CDate::CheckHour (uint32_t hour) [static]`

Checks if an hour value is valid

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_ERROR_INVALID_HOUR, and BRATHL_SUCCESS.

6.35.3.7 `int32_t brathl::CDate::CheckMinute (uint32_t minute) [static]`

Checks if a minute is valid

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_ERROR_INVALID_MINUTE**, and **BRATHL_SUCCESS**.

6.35.3.8 `int32_t brathl::CDate::CheckMonth (uint32_t month) [static]`

Checks if a month value is valid

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_ERROR_INVALID_MONTH**, and **BRATHL_SUCCESS**.

Referenced by `DayOfYear()`.

6.35.3.9 `int32_t brathl::CDate::CheckMuSecond (uint32_t muSecond) [static]`

Checks if a month value is valid

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_ERROR_INVALID_MUSECOND**, and **BRATHL_SUCCESS**.

6.35.3.10 `int32_t brathl::CDate::CheckSecond (uint32_t second) [static]`

Checks if a second value is valid

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_ERROR_INVALID_SECOND**, and **BRATHL_SUCCESS**.

6.35.3.11 `int32_t brathl::CDate::CheckYear (uint32_t year) [static]`

Checks if a year value is valid year have to be >= internal reference year (1950)

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_ERROR_INVALID_YEAR**, **BRATHL_SUCCESS**, and `m_internalRefYear`.

Referenced by `DayOfYear()`.

6.35.3.12 `int32_t brathl::CDate::ConstructDate (const brathl_refDate refDate)`

Converts a date whose value corresponds to the date reference enumeration

Parameters

<i>refDate</i>	[in]: date reference - see brathl_refDate (p. 372))
----------------	--

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_ERROR_INVALID_DATE_REF, brathl_refDateUser1, brathl_refDateUser2, BRATHL_SUCCESS, REF19500101, REF19580101, REF19850101, REF19900101, REF20000101, REFUSER1, and REFUSER2.

6.35.3.13 `int32_t brathl::CDate::Convert2DecimalJulian (double & julian, const brathl_refDate refDate = REF19500101)`

Converts the date value into a decimal julian day

Parameters

<i>julian</i>	[out]: decimal julian day (can be < 0)
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS, and m_secInDay.

Referenced by brathl_DSM2Julian(), brathl_Seconds2Julian(), brathl_YMDHMSM2Julian(), and brathl::CMission::Convert().

6.35.3.14 `int32_t brathl::CDate::Convert2DMM (int32_t & days, int32_t & milliSeconds, int32_t & muSeconds, const brathl_refDate refDate = REF19500101)`

Converts the date value into a number of days, milliseconds, microseconds

Parameters

<i>days</i>	[out]: number of days (can be < 0)
<i>milliSeconds</i>	[out]: number of milliseconds
<i>muSeconds</i>	[out]: number of microseconds
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS, m_minutesInDay, m_secInDay, and m_secInMinute.

6.35.3.15 `int32_t brathl::CDate::Convert2DMM (double & days, double & milliSeconds, double & muSeconds, const brathl_refDate refDate = REF19500101)`

Converts the date value into a number of days, milliseconds, microseconds

Parameters

<i>days</i>	[out]: number of days (can be < 0)
<i>milliSeconds</i>	[out]: number of milliseconds
<i>muSeconds</i>	[out]: number of microseconds
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS.

6.35.3.16 `int32_t brathl::CDate::Convert2DSM (int32_t & days, int32_t & seconds, int32_t & muSeconds, const brathl_refDate refDate = REF19500101)`

Converts the date value into a number of days, seconds, microseconds

Parameters

<i>days</i>	[out]: number of days (can be < 0)
<i>seconds</i>	[out]: number of seconds
<i>muSeconds</i>	[out]: number of microseconds
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS, m_minutesInDay, m_secInDay, and m_secInMinute.

Referenced by brathl_Julian2DSM(), brathl_Seconds2DSM(), and brathl_YMDHMSM2DSM().

6.35.3.17 `int32_t brathl::CDate::Convert2DSM (double & days, double & seconds, double & muSeconds, const brathl_refDate refDate = REF19500101)`

Converts the date value into a number of days, seconds, microseconds

Parameters

<i>days</i>	[out]: number of days (can be < 0)
<i>seconds</i>	[out]: number of seconds
<i>muSeconds</i>	[out]: number of microseconds
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS.

6.35.3.18 `int32_t brathl::CDate::Convert2Second (double & seconds, const brathl_refDate refDate = REF19500101)`

Converts the date value into a decimal number of seconds

Parameters

<i>seconds</i>	[out]: decimal number of seconds day (can be < 0)
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS, and Value().

Referenced by brathl_DSM2Seconds(), brathl_Julian2Seconds(), and brathl_YMDHMSM2Seconds().

6.35.3.19 `int32_t brathl::CDate::Convert2SM (int32_t & seconds, int32_t & muSeconds, const brathl_refDate refDate = REF19500101)`

Converts the date value into a number of seconds, microseconds

Parameters

<i>seconds</i>	[out]: number of milliseconds (can be < 0)
<i>muSeconds</i>	[out]: number of microseconds
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS, and m_secInMinute.

6.35.3.20 `int32_t brathl::CDate::Convert2SM (double & seconds, double & muSeconds, const brathl_refDate refDate = REF19500101)`

Converts the date value into a number of seconds, microseconds

Parameters

<i>seconds</i>	[out]: number of milliseconds (can be < 0)
<i>muSeconds</i>	[out]: number of microseconds
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS.

6.35.3.21 `int32_t brathl::CDate::Convert2YMDHMSM (uint32_t & year, uint32_t & month, uint32_t & day, uint32_t & hour, uint32_t & minute, uint32_t & second, uint32_t & muSecond)`

Converts the date value into year, month, day, hour, minute, second, microsecond

Parameters

<i>year</i>	[out]: year
<i>month</i>	[out]: month
<i>day</i>	[out]: day
<i>hour</i>	[out]: hour
<i>minute</i>	[out]: minute
<i>second</i>	[out]: second
<i>muSecond</i>	[out]: microsecond

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS, m_daysOfYear, m_internalRefYear, m_minutesInDay, and m_minutesInHour.

Referenced by brathl_Cycle2YMDHMSM(), brathl_DSM2YMDHMSM(), brathl_Julian2YMDHMSM(), brathl_NowYMDHMSM(), and brathl_Seconds2YMDHMSM().

6.35.3.22 `double brathl::CDate::CvDate (const char * strDate) [static]`

Convert a date string to a number of seconds since internal reference year (ie 1950) Allowed format are :

- YYYY-MM-DD HH:MM:SS.MS string
- a julian string (format:positive 'Days Seconds Microseconds' or positive decimal julian day) For julian string, it

can contain its date reference at the end by specifying where YYYY the reference year. If no date reference is specified the default date reference is used.

Parameters

<i>strDate</i>	: date string
----------------	---------------

Returns

number of seconds since internal reference year (ie 1950)

References BRATHL_INCONSISTENCY_ERROR, BRATHL_SUCCESS, brathl::CTools::Format(), SetDate(), and Value().

6.35.3.23 uint32_t brathl::CDate::DayOfYear (uint32_t year, uint32_t month, uint32_t day) [static]

Retrieves the day of a year if year is not valid, methods force the value to the internal reference year (1950) if month is not valid, methods force the value to 1 day value is not check

Parameters

<i>year</i>	[in]: year
<i>month</i>	[in]: month of year
<i>day</i>	[in]: day of the month

Returns

the day of year

References BRATHL_SUCCESS, CheckMonth(), CheckYear(), LeapYearIndex(), m_daysOfYear, and m_internal-RefYear.

Referenced by brathl_DayOfYear().

6.35.3.24 uint32_t brathl::CDate::DayOfYear (CDate & date) [static]

Retrieves the day of year of a **CDate** (p. 173) object

Parameters

<i>date</i>	[in]: date
-------------	------------

Returns

the day of year

References GetDay(), GetMonth(), LeapYearIndex(), and m_daysOfYear.

6.35.3.25 uint32_t brathl::CDate::DayOfYear ()

Retrieves the day of year of the date object

Returns

the day of year

6.35.3.26 int32_t brathl::CDate::GetDaysInMonth (const uint32_t month, const uint32_t year, uint32_t & nbDaysInMonth) [static]

Retrieves the number of days in a month, according to a year and a month

Parameters

<i>month</i>	[in] : month
<i>year</i>	[in] : year
<i>nbDaysInMonth[out]</i>	: number of days in the month

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS.

6.35.3.27 uint32_t brathl::CDate::HowManyLeapYear (const uint32_t year)

Computes the number of leap years since a year

Parameters

<i>year</i>	[in]: year
-------------	------------

Returns

number of leap years

References IsLeapYear(), and m_internalRefYear.

6.35.3.28 void brathl::CDate::InitDateZero ()

Initializes a **CDate** (p. 173) object to 0

6.35.3.29 bool brathl::CDate::IsDefaultValue ()

Tests the internal value to the default value

Returns

true if default value, otherwise false

Referenced by brathl::CDatePeriod::Intersect().

6.35.3.30 bool brathl::CDate::IsLeapYear (const uint32_t year) [static]

Testd if the year is a leap year

Parameters

<i>year</i>	[in]: year to test
-------------	--------------------

Returns

true if the year is a leap year, otherwise false

6.35.3.31 bool brathl::CDate::IsLeapYear ()

Tests if the year of the date object is a leap year

Returns

true if the year of the date object is a leap year, otherwise false

Referenced by `HowManyLeapYear()`, and `LeapYearIndex()`.

6.35.3.32 `int32_t brathl::CDate::LeapYearIndex (const uint32_t year) [static]`

Retrieves the index of the **m_daysOfYear** (p. 189) or **m_daysInMonth** (p. 189) arrays in accordance with the year (leap year or not)

Parameters

<i>year</i>	[in]: year to test
-------------	--------------------

Returns

0 if year is a leap year, otherwise 1

References `IsLeapYear()`.

Referenced by `DayOfYear()`.

6.35.3.33 `int32_t brathl::CDate::LeapYearIndex ()`

Retrieve sthe index of the `daysOfYear` or `daysInMonth` arrays in accordance with the year of the date object (leap year or not)

Returns

0 if year of the date object is a leap year, otherwise 1

Referenced by `DayOfYear()`.

6.35.3.34 `double brathl::CDate::operator+ (CDate & d) [inline]`

Plus operator redefinition Computes the addition of two dates, the result is expressed in a decimal number of seconds

References `Value()`.

6.35.3.35 `double brathl::CDate::operator- (CDate & d) [inline]`

Minus operator redefinition Computes the difference between two dates, the result is expressed in a decimal number of seconds

References `Value()`.

6.35.3.36 `bool brathl::CDate::operator< (CDate & d) [inline]`

Comparison operators

References `Value()`.

6.35.3.37 `const CDate & brathl::CDate::operator= (const CDate & date)`

Assigns a new value to the **CDate** (p. 173) object, with a **CDate** (p. 173) object

6.35.3.38 `const CDate & brathl::CDate::operator= (const char * strDate)`

Assigns a new value to the **CDate** (p. 173) object, with a date string (format: YYYY-MM-DD HH:MN:SS.MS)

6.35.3.39 `const CDate & brathl::CDate::operator= (double seconds)`

Assigns a new value to the **CDate** (p. 173) object, with a number of seconds since 1950-01-01

6.35.3.40 `const CDate & brathl::CDate::operator= (const brathl_refDate refDate)`

Assigns a new value to the **CDate** (p. 173) object, with a reference date

6.35.3.41 `int32_t brathl::CDate::SetDate (const char * strDate)`

Sets date value from a string Allowed format are :

- YYYY-MM-DD HH:MM:SS.MS string
- a julian string (format:positive 'Days Seconds Microseconds' or positive decimal julian day) For julian string, it can contain its date reference at the end by specifying where YYYY the reference year. If no date reference is specified the default date reference is used.

Parameters

<i>strDate</i>	: date string
----------------	---------------

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_ERROR_INVALID_DATE, and BRATHL_SUCCESS.

Referenced by brathl_DayOfYear(), brathl_DiffDSM(), brathl_DiffJulian(), brathl_DiffYMDHMSM(), brathl_DSM2-Julian(), brathl_DSM2Seconds(), brathl_DSM2YMDHMSM(), brathl_Julian2DSM(), brathl_Julian2Seconds(), brathl_Julian2YMDHMSM(), brathl_Seconds2DSM(), brathl_Seconds2Julian(), brathl_Seconds2YMDHMSM(), brathl_YMDHMSM2Cycle(), brathl_YMDHMSM2DSM(), brathl_YMDHMSM2Julian(), brathl_YMDHMSM2Seconds(), and CvDate().

6.35.3.42 `int32_t brathl::CDate::SetDate (const brathl_DateYMDHMSM & date)`

Sets date value from a **brathl_DateYMDHMSM** (p. 371) structure

Parameters

<i>date</i>	[in]: brathl_DateYMDHMSM (p. 371) structure date
-------------	---

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS.

6.35.3.43 `int32_t brathl::CDate::SetDate (const brathl_DateDSM & date)`

Sets date value from a **brathl_DateDSM** (p. 371) structure

Parameters

<i>date</i>	[in]: brathl_DateDSM (p. 371) structure date
-------------	---

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_ERROR_INVALID_DSM**, **BRATHL_SUCCESS**, `_structDateDSM::days`, `_structDateDSM::muSeconds`, `_structDateDSM::refDate`, and `_structDateDSM::seconds`.

6.35.3.44 `int32_t brathl::CDate::SetDate (const uint32_t days, const uint32_t seconds, const uint32_t muSeconds, const brathl_refDate refDate = REF19500101)`

Sets date value from year, month, day, hour, minute, second, microsecond

Parameters

<i>days</i>	[in]: number of days
<i>seconds</i>	[in]: number of seconds
<i>muSeconds</i>	[in]: number of microseconds
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

6.35.3.45 `int32_t brathl::CDate::SetDate (const brathl_DateSecond & date)`

Sets date value from a **brathl_DateSecond** (p. 371) structure

Parameters

<i>date</i>	[in]: brathl_DateSecond (p. 371) structure date
-------------	--

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References `_structDateSecond::nbSeconds`, and `_structDateSecond::refDate`.

6.35.3.46 `int32_t brathl::CDate::SetDate (const brathl_DateJulian & date)`

Sets date value from a **brathl_DateJulian** (p. 371) structure

Parameters

<i>date</i>	[in]: brathl_DateJulian (p. 371) structure date
-------------	--

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References `_structDateJulian::julian`, and `_structDateJulian::refDate`.

6.35.3.47 `int32_t brathl::CDate::SetDate (const uint32_t year, const uint32_t month = 1, const uint32_t day = 1, const uint32_t hour = 0, const uint32_t minute = 0, const uint32_t second = 0, const uint32_t muSecond = 0)`

Sets date value from year, month, day, hour, minute, second, microsecond

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**.

6.35.3.48 `int32_t brathl::CDate::SetDate (const double dateSeconds, brathl_refDate refDate = REF19500101)`

Sets date value from a decimal number of seconds

Parameters

<i>dateSeconds</i>	[in]: decimal number of seconds
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_ERROR_INVALID_YEAR, BRATHL_SUCCESS, and m_secInMinute.

6.35.3.49 `int32_t brathl::CDate::SetDateJulian (const double dateJulian, brathl_refDate refDate = REF19500101)`

Sets date value from a decimal julian day

Parameters

<i>dateJulian</i>	[in]: decimal julian day
<i>refDate</i>	[in]: date reference (default value is REF19500101 - see brathl_refDate (p. 372))

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_ERROR_INVALID_YEAR, BRATHL_SUCCESS, m_minutesInDay, m_secInMinute, and ValueJulian().

Referenced by brathl::CMission::Convert().

6.35.3.50 `int32_t brathl::CDate::SetDateNow ()`

Sets the date object to the current time

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References BRATHL_SUCCESS.

Referenced by brathl_NowYMDHMSM().

6.35.3.51 `void brathl::CDate::SetDefaultValue ()`

Sets internal value to the default value

6.35.3.52 `int32_t brathl::CDate::SubtractDays (uint32_t days)`

Subtracts a number of day from the date object

Parameters

<i>days</i>	[in]: number of days to subtract (if < 0, a addition operation is performed)
-------------	--

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Date error codes** (p. 18))

References **BRATHL_SUCCESS**, and **m_minutesInDay**.

6.35.4 Member Data Documentation

6.35.4.1 `const uint32_t brathl::CDate::m_daysInMonth` [static]

Initial value:

```
{
    { 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 },
    { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 }
}
```

Array[i,j] of number of days in month i : 0 corresponds to a leap year, 1 corresponds to a non-leap year j : index of the month

6.35.4.2 `const uint32_t brathl::CDate::m_daysOfYear` [static]

Initial value:

```
{
    { 0, 31, 60, 91, 121, 152, 182, 213, 244, 274, 305, 335 },
    { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334 }
}
```

Array[i,j] of day of year i : 0 corresponds to a leap year, 1 corresponds to a non-leap year j : index of the month

Referenced by **Convert2YMDHMSM()**, and **DayOfYear()**.

6.35.4.3 `const uint32_t brathl::CDate::m_internalRefYear = 1950` [static]

Internal reference year (1950)

Referenced by **CheckYear()**, **Convert2YMDHMSM()**, **DayOfYear()**, and **HowManyLeapYear()**.

6.35.4.4 `const double brathl::CDate::m_minutesInDay = 1440.0` [static]

Number of minutes in a day

Referenced by **AddDays()**, **Convert2DMM()**, **Convert2DSM()**, **Convert2YMDHMSM()**, **SetDateJulian()**, and **SubtractDays()**.

6.35.4.5 `const double brathl::CDate::m_minutesInHour = 60.0` [static]

Number of minutes in an hour

Referenced by **Convert2YMDHMSM()**.

6.35.4.6 `const double brathl::CDate::m_secInDay = 86400.0` [static]

Number of seconds in a day

Referenced by **Convert2DecimalJulian()**, **Convert2DMM()**, and **Convert2DSM()**.

6.35.4.7 `const double brathl::CDate::m_secInHour = 3600.0` [static]

Number of seconds in an hour

6.35.4.8 `const double bratl::CDate::m_secInMinute = 60.0` [static]

Number of seconds in a minute

Referenced by `Convert2DMM()`, `Convert2DSM()`, `Convert2SM()`, `SetDate()`, and `SetDateJulian()`.

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

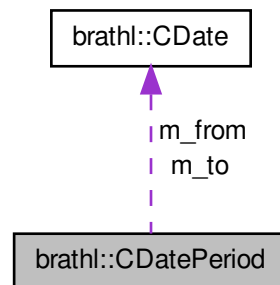
- `Date.h`
- `Date.cpp`

6.36 bratl::CDatePeriod Class Reference

```
#include <DatePeriod.h>
```

Inherits `bratl::CBratObject`.

Collaboration diagram for `bratl::CDatePeriod`:



Public Member Functions

- string **AsString** (const string &format="", bool withMuSecond=false)
- **CDatePeriod** ()
Empty CDatePeriod (p. 190) ctor.
- **CDatePeriod** (CDatePeriod &datePeriod)
- **CDatePeriod** (CDate &from, CDate &to)
- **CDatePeriod** (const string &from, const string &to)
- **CDatePeriod** (double from, double to)
- **CDatePeriod** (const CStringArray &array)
- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- string **GetAsText** (const string &delimiter=CDatePeriod::m_delimiter)
- string **GetFormat** ()
- CDate & **GetFrom** ()
- string **GetFromAsText** ()
- CDate & **GetTo** ()
- string **GetToAsText** ()
- bool **GetWithMuSecond** ()
- bool **Intersect** (CDatePeriod &datePeriod, CDatePeriod &intersect)
- bool **Intersect** (CDate &otherFrom, CDate &otherTo, CDatePeriod &intersect)

- bool **IsDefaultValue** ()
- const **CDatePeriod** & **operator=** (**CDatePeriod** &datePeriod)
- void **Set** (**CDate** &from, **CDate** &to)
- void **Set** (const string &from, const string &to)
- void **Set** (double from, double to)
- void **Set** (const CStringArray &array)
- void **Set** (**CDatePeriod** &datePeriod)
- void **SetDefaultValue** ()
- void **SetFormat** (const string &value)
- void **SetFrom** (**CDate** &from)
- void **SetFrom** (const string &strDate)
- void **SetTo** (**CDate** &to)
- void **SetTo** (const string &strDate)
- void **SetWithMuSecond** (bool value)
- bool **Union** (**CDatePeriod** &datePeriod)
- bool **Union** (**CDate** &otherFrom, **CDate** &otherTo)
- bool **Union** (**CDatePeriod** &datePeriod, **CDatePeriod** &unionDate)
- bool **Union** (**CDate** &otherFrom, **CDate** &otherTo, **CDatePeriod** &unionDate)
- virtual ~**CDatePeriod** ()

Destructor.

Static Public Attributes

- static const string **m_delimiter** = "/"

Protected Member Functions

- void **Adjust** ()
- void **Init** ()

Protected Attributes

- string **m_format**
- **CDate** **m_from**
- **CDate** **m_to**
- bool **m_withMuSecond**

6.36.1 Detailed Description

Date interval management class.

Version

1.0

6.36.2 Constructor & Destructor Documentation

6.36.2.1 brathl::CDatePeriod::CDatePeriod (**CDatePeriod** & *datePeriod*)

Copy constructor.

Parameters

<i>datePeriod</i>	period to set
-------------------	---------------

6.36.2.2 bratl::CDatePeriod::CDatePeriod (CDate & *from*, CDate & *to*)

Constructor.

Parameters

<i>from</i>	start date
<i>to</i>	end date

6.36.2.3 bratl::CDatePeriod::CDatePeriod (const string & *from*, const string & *to*)

Constructor.

Parameters

<i>from</i>	start date
<i>to</i>	end date

6.36.2.4 bratl::CDatePeriod::CDatePeriod (double *from*, double *to*)

Constructor.

Parameters

<i>from</i>	start date (number of seconds since 1950-01-01)
<i>to</i>	end date (number of seconds since 1950-01-01)

6.36.2.5 bratl::CDatePeriod::CDatePeriod (const CStringArray & *array*)

Constructor from a array that contains start date as string, end date as string

Parameters

<i>array</i>	start and end dates
--------------	---------------------

6.36.3 Member Function Documentation

6.36.3.1 CDate& bratl::CDatePeriod::GetFrom () [inline]

Gets start date

Returns

start date

Referenced by Intersect(), and Set().

6.36.3.2 CDate& bratl::CDatePeriod::GetTo () [inline]

Gets end date

Returns

end date

Referenced by Intersect(), and Set().

6.36.3.3 bool bratl::CDatePeriod::Intersect (CDatePeriod & *datePeriod*, CDatePeriod & *intersect*)

Create the intersection of this date period with the given one

Parameters

<i>datePeriod</i>	intersect with this
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

References GetFrom(), and GetTo().

6.36.3.4 bool brathl::CDatePeriod::Intersect (CDate & otherFrom, CDate & otherTo, CDatePeriod & intersect)

Create the intersection of this date period with the given one

Parameters

<i>otherFrom</i>	start date intersect with this
<i>otherTo</i>	end date intersect with this
<i>intersect</i>	intersection period

Returns

true, or false if there is no intersection

References brathl::CDate::IsDefaultValue(), SetFrom(), and SetTo().

6.36.3.5 bool brathl::CDatePeriod::IsDefaultValue ()

Tests whether date period have been initialized or not

Returns

true if not initialized

6.36.3.6 const CDatePeriod & brathl::CDatePeriod::operator= (CDatePeriod & datePeriod)

Assigns a new value to the **CDatePeriod** (p. 190) object, with a **CDatePeriod** (p. 190) object

6.36.3.7 void brathl::CDatePeriod::Set (CDate & from, CDate & to)

Sets date period from start and end date

Parameters

<i>from</i>	start date
<i>to</i>	end date

6.36.3.8 void brathl::CDatePeriod::Set (const string & from, const string & to)

Sets date period from start and end date

Parameters

<i>from</i>	start date
<i>to</i>	end date

6.36.3.9 void brathl::CDatePeriod::Set (const CStringArray & array)

Sets a date period from a array that contains start date as string, end date as string

Parameters

<i>array</i>	start and end dates
--------------	---------------------

6.36.3.10 void brathl::CDatePeriod::Set (CDatePeriod & datePeriod)

Sets date period from another one

Parameters

<i>datePeriod</i>	period to set
-------------------	---------------

References GetFrom(), and GetTo().

6.36.3.11 void brathl::CDatePeriod::SetDefaultValue ()

Sets internal value to the default value (uninitialized)

6.36.3.12 void brathl::CDatePeriod::SetFrom (CDate & from)

Sets start date

Parameters

<i>to</i>	start date
-----------	------------

Referenced by Intersect().

6.36.3.13 void brathl::CDatePeriod::SetFrom (const string & strDate)

Sets start date

Parameters

<i>to</i>	start date
-----------	------------

References BRATHL_SUCCESS, BRATHL_SYNTAX_ERROR, and brathl::CTools::Format().

6.36.3.14 void brathl::CDatePeriod::SetTo (CDate & to)

Sets end date

Parameters

<i>to</i>	end date
-----------	----------

Referenced by Intersect().

6.36.3.15 void brathl::CDatePeriod::SetTo (const string & strDate)

Sets end date

Parameters

<i>to</i>	end date
-----------	----------

References BRATHL_SUCCESS, BRATHL_SYNTAX_ERROR, and brathl::CTools::Format().

6.36.4 Member Data Documentation

6.36.4.1 CDate brathl::CDatePeriod::m_from [protected]

Start date

6.36.4.2 CDate brathl::CDatePeriod::m_to [protected]

End date

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

- DatePeriod.h
- DatePeriod.cpp

6.37 brathl::CDoubleArray Class Reference

```
#include <List.h>
```

Inherited by brathl::CDoubleArrayOb.

Public Member Functions

- **CDoubleArray** ()
Empty CDoubleArray (p. 195) ctor.
- **CDoubleArray** (const **CDoubleArray** &vect)
- const double * **data** () const
- virtual void **Dump** (ostream &fOut=cerr) const
Dump function.
- virtual bool **Erase** (CDoubleArray::iterator it)
- virtual int32_t **FindIndex** (double value) const
- void **GetRange** (double &min, double &max)
- virtual void **Insert** (double *data, int32_t size)
- virtual void **Insert** (int32_t *data, int32_t size)
- virtual void **Insert** (uint32_t *data, int32_t size)
- virtual void **Insert** (const **CDoubleArray** &vect, bool bEnd=true)
- virtual void **Insert** (const **CDoubleArray** &vect, int32_t first, int32_t last, bool bEnd=true)
- virtual void **Insert** (const **CUInt8Array** &vect, bool bEnd=true)
- virtual void **Insert** (const **CInt8Array** &vect, bool bEnd=true)
- virtual void **Insert** (const **CInt16Array** &vect, bool bEnd=true)
- virtual void **Insert** (const **CIntArray** &vect, bool bEnd=true)
- virtual void **Insert** (const **CFloatArray** &vect, bool bEnd=true)
- virtual void **Insert** (const CStringArray &vect, bool bEnd=true)
- virtual void **Insert** (const string &vect, const string &delim=",", bool bEnd=true)
- virtual void **Insert** (const double value)
- virtual void **Insert** (const int32_t value)
- virtual void **Insert** (const uint32_t value)
- virtual void **Insert** (const int16_t value)
- virtual void **Insert** (const uint16_t value)
- virtual void **Insert** (const int8_t value)
- virtual void **Insert** (const uint8_t value)
- virtual CDoubleArray::iterator **InsertAt** (CDoubleArray::iterator where, const double value)
- virtual CDoubleArray::iterator **InsertAt** (int32_t index, const double value)
- virtual bool **Intersect** (const **CDoubleArray** &array, **CDoubleArray** &intersect) const
- virtual bool **operator!=** (const **CDoubleArray** &vect)

- virtual const **CDoubleArray** & **operator=** (const **CDoubleArray** &vect)
- virtual bool **operator==** (const **CDoubleArray** &vect)
- virtual void **RemoveAll** ()
- virtual CDoubleArray::iterator **ReplaceAt** (CDoubleArray::iterator where, const double value)
- virtual CDoubleArray::iterator **ReplaceAt** (int32_t index, const double value)
- double * **ToArray** ()
- virtual string **ToString** (const string &delim="," , bool useBracket=true) const
- virtual ~**CDoubleArray** ()

Destructor.

6.37.1 Detailed Description

An array (vector) of double management class.

Version

1.0

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

- List.h
- List.cpp

6.38 brathl::CDoubleMap Class Reference

```
#include <List.h>
```

Public Member Functions

- **CDoubleMap** ()
CDoubleMap (p. 196) ctor.
- virtual void **Dump** (ostream &fOut=cerr) const
Dump function.
- virtual bool **Erase** (CDoubleMap::iterator it)
- virtual bool **Erase** (const string &key)
- virtual double **Exists** (const string &key) const
- virtual double **Insert** (const string &key, double value, bool withExcept=true)
- virtual double **operator[]** (const string &key)
- virtual void **RemoveAll** ()
- virtual ~**CDoubleMap** ()
CDoubleMap (p. 196) dtor.

6.38.1 Detailed Description

a set of double value management classes.

Version

1.0

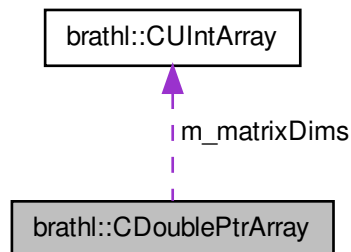
The documentation for this class was generated from the following files:

- List.h
- List.cpp

6.39 brathl::CDoublePtrArray Class Reference

```
#include <List.h>
```

Collaboration diagram for brathl::CDoublePtrArray:



Public Member Functions

- **CDoublePtrArray** (bool bDelete=true)
Empty CDoublePtrArray (p. 197) ctor.
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CDoublePtrArray::iterator it)
- virtual bool **Erase** (int32_t index)
- bool **GetDelete** ()
- uint32_t **GetMatrixDim** (uint32_t row)
- **CUIntArray** * **GetMatrixDims** ()
- size_t **GetMatrixNumberOfDims** ()
- virtual void **Insert** (DoublePtr ob)
- virtual CDoublePtrArray::iterator **InsertAt** (CDoublePtrArray::iterator where, DoublePtr ob)
- DoublePtr **NewMatrix** (double initialValue=CTools::m_defaultValueDOUBLE)
- virtual bool **PopBack** ()
- virtual void **RemoveAll** ()
- virtual CDoublePtrArray::iterator **ReplaceAt** (CDoublePtrArray::iterator where, DoublePtr ob)
- void **SetDelete** (bool value)
- void **SetMatrixDims** (const **CUIntArray** &matrixDims)
- virtual ~**CDoublePtrArray** ()
Destructor.

Protected Member Functions

- void **Delete** (DoublePtr matrix)

Protected Attributes

- bool **m_bDelete**
- **CUIntArray** **m_matrixDims**

6.39.1 Detailed Description

An array (vector) of duple pointer management class.

Version

1.0

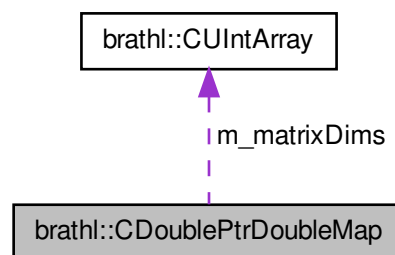
The documentation for this class was generated from the following files:

- List.h
- List.cpp

6.40 brathl::CDoublePtrDoubleMap Class Reference

```
#include <List.h>
```

Collaboration diagram for brathl::CDoublePtrDoubleMap:



Public Member Functions

- **CDoublePtrDoubleMap** (bool bDelete=true)
CDoublePtrDoubleMap (p. 198) ctor.
- **CDoublePtrDoubleMap** (const **CUIIntArray** &matrixDims, bool bDelete=true)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump function.
- virtual bool **Erase** (CDoublePtrDoubleMap::iterator it)
- virtual bool **Erase** (double key)
- virtual DoublePtr * **Exists** (double key) const
- bool **GetDelete** ()
- virtual void **GetKeys** (CDoubleArray &keys, bool bRemoveAll=true)
- uint32_t **GetMatrixColDim** (uint32_t row)
- **CUIIntArray** * **GetMatrixDims** ()
- size_t **GetMatrixNumberOfRows** () const
- virtual DoublePtr * **Insert** (double key, DoublePtr *ob, bool withExcept=true)
- virtual DoublePtr * **Insert** (double key, double initialValue=**CTools::m_defaultValueDOUBLE**)
- DoublePtr * **NewMatrix** (double initialValue=**CTools::m_defaultValueDOUBLE**)
- virtual DoublePtr * **operator[]** (double key)
- virtual void **RemoveAll** ()

- bool **RenameKey** (double oldKey, double newKey)
- void **SetDelete** (bool value)
- void **SetMatrixDims** (const **CUIntArray** &matrixDims)
- virtual **~CDoublePtrDoubleMap** ()

CDoublePtrDoubleMap (p. 198) *dtor.*

Protected Member Functions

- void **Delete** (DoublePtr *matrix)

Protected Attributes

- bool **m_bDelete**
- **CUIntArray** **m_matrixDims**

6.40.1 Detailed Description

a set of a non rectangular matrix of double management classes.

Version

1.0

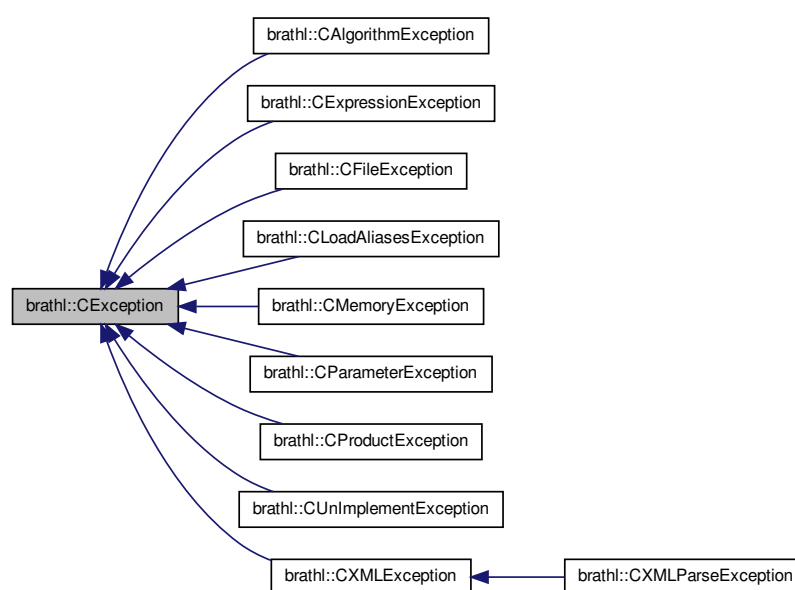
The documentation for this class was generated from the following files:

- List.h
- List.cpp

6.41 brathl::CException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for brathl::CException:



Public Member Functions

- **CException** ()
Empty CException (p. 199) ctor.
- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- int32_t **error** ()
- string **GetMessage** ()
- virtual const char * **TypeOf** () const
- virtual const char * **what** () const throw ()
- virtual ~**CException** () throw ()
Destructor.
- **CException** (const string &message, int32_t errcode)

Protected Attributes

- int32_t **m_errcode**
- string **m_message**

6.41.1 Detailed Description

Exception management class.

Version

1.0

6.41.2 Constructor & Destructor Documentation

6.41.2.1 brathl::CException::CException (const string & message, int32_t errcode)

Creates a new **CException** (p. 199) object.

Parameters

<i>message</i>	[in] : error message
<i>errcode</i>	[in] : error code

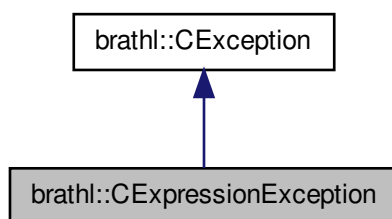
The documentation for this class was generated from the following files:

- **Exception.h**
- **Exception.cpp**

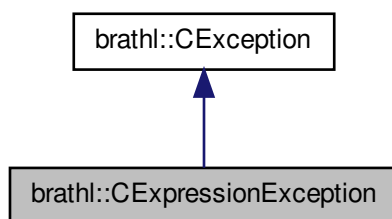
6.42 brathl::CExpressionException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for bratl::CExpressionException:



Collaboration diagram for bratl::CExpressionException:



Public Member Functions

- **CExpressionException** ()
*Empty **CExpressionException** (p. 200) ctor.*
- **CExpressionException** (const string &message, int32_t errcode, const string &expression="")
- virtual const char * **TypeOf** () const
Identification of exception (human readable)
- virtual ~**CExpressionException** () throw ()
Destructor.

Additional Inherited Members

6.42.1 Detailed Description

Expression Exception management class.

Version

1.0

6.42.2 Constructor & Destructor Documentation

6.42.2.1 brathl::CExpressionException::CExpressionException (const string & message, int32_t errcode, const string & expression = " ")

Creates a new **CParameterException** (p. 278) object.

Parameters

<i>message</i>	[in] : error message
<i>errcode</i>	[in] : error code
<i>expression</i>	[in] : expression being compiled

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

- **Exception.h**
- Exception.cpp

6.43 brathl::CExpressionValue Class Reference

```
#include <Expression.h>
```

Inherits brathl::CBratObject.

Public Member Functions

- string **AsString** (const CUnit &Unit=CUnit(""), const string Format="", bool dateAsPeriod=false) const
- **CExpressionValue** (double FloatValue=**CTools::m_defaultValueDOUBLE**)
- **CExpressionValue** (const vector< double > &FloatValues)
- **CExpressionValue** (const string &StrValue)
- **CExpressionValue** (ExpressionValueType Type, **ExpressionValueDimensions** &Dimensions, double *Value, bool MakeCopy=true)
- **CExpressionValue** (ExpressionValueType type, **ExpressionValueDimensions** &dimensions, const **C-DoubleArray** &value)
- **CExpressionValue** (const **CExpressionValue** &Copy)
- **CExpressionValue** (ExpressionCallableFunction1 &Function, bool IsNumeric, **CExpressionValue** &Parameter1)
- **CExpressionValue** (ExpressionCallableFunctionStrToStr1 &Function, **CExpressionValue** &Parameter1)
- **CExpressionValue** (ExpressionCallableFunctionStrToFlt1 &Function, **CExpressionValue** &Parameter1)
- **CExpressionValue** (ExpressionCallableFunction2 &Function, bool IsNumeric, **CExpressionValue** &Parameter1, **CExpressionValue** &Parameter2)
- **CExpressionValue** (ExpressionCallableFunction3 &Function, bool IsNumeric, **CExpressionValue** &Parameter1, **CExpressionValue** &Parameter2, **CExpressionValue** &Parameter3)
- **CExpressionValue** (ExpressionCallableFunctionAlgoN &function, const char *functionName, CVectorBratAlgorithmParam &arg)
- **CExpressionValue** (ExpressionCallableFunctionBratAlgoBaseN &function, **CBratAlgorithmBase** *algo, CVectorBratAlgorithmParam &arg)
- double **Compare** (**CExpressionValue** &WithWhat)
- void **DeleteValue** ()
- void **Dump** (ostream &fOut=cerr)
- const **ExpressionValueDimensions** & **GetDimensions** () const
- string **GetDimensionsAsString** ()
- string **GetName** ()
- uint32_t **GetNbDimensions** () const
- uint32_t **GetNbValues** () const
- string **GetString** () const

- ExpressionValueType **GetType** () const
- double **GetValue** (uint32_t index) const
- double **GetValue** (uint32_t i, uint32_t j) const
- double * **GetValues** () const
- bool **HasValue** ()
- int32_t **IsTrue** ()
- **CExpressionValue** & **operator=** (const **CExpressionValue** &Copy)
- **CExpressionValue** & **operator=** (const string &String)
- **CExpressionValue** & **operator=** (double value)
- **CExpressionValue** & **operator=** (const vector< double > &Vector)
- void **Set** (const **CExpressionValue** &Copy)
- void **SetName** (const string &value)
- void **SetNewValue** (ExpressionValueType type, uint32_t *dims, uint32_t nbDims, double *value, bool makeCopy=true)
- void **SetNewValue** (ExpressionValueType Type, **ExpressionValueDimensions** &Dimensions, double *Value, bool MakeCopy=true)
- void **SetNewValue** (**CDoubleArray** &vect, bool makeCopy=true)
- void **SetNewValue** (**CObDoubleMap** &mp, bool makeCopy=true)
- void **SetNewValue** (**CDoublePtrDoubleMap** &mp, bool makeCopy=true)
- void **SetNewValue** (double *dataValue, uint32_t nbValues, bool makeCopy=true)

Static Public Member Functions

- static **CExpressionValue** * **GetExpressionValue** (CBratObject *ob, bool withExcept=true)

6.43.1 Detailed Description

Expression management classes.

Version

1.0

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

- Expression.h
- Expression.cpp

6.44 brathl::CExternalFilesAvisoGrid Class Reference

```
#include <ExternalFilesAvisoGrid.h>
```

Inherits brathl::CExternalFilesNetCDFCF.

Inherited by brathl::CExternalFilesDotGrid, and brathl::CExternalFilesMercatorDotGrid.

Public Member Functions

- **CExternalFilesAvisoGrid** (const string &Name="")
- virtual void **GetValue** (const string &Name, **CExpressionValue** &Value, const string &WantedUnit)
- virtual void **GetValue** (const string &name, double &value, const string &wantedUnit)
- virtual bool **NextRecord** ()
- virtual bool **PrevRecord** ()
- virtual void **Rewind** ()

Static Public Member Functions

- static string **TypeOf** ()

Static Public Attributes

- static const string **m_INTERNAL_DEPTH_DIM_NAME** = "GridDepth"
- static const string **m_INTERNAL_LAT_DIM_NAME** = "NbLatitudes"
- static const string **m_INTERNAL_LATLON_DIM_NAME** = "LatLon"
- static const string **m_INTERNAL_LON_DIM_NAME** = "NbLongitudes"
- static const string **m_LAT_DIM_NAME** = "Latitude"
- static const string **m_LATLONMIN_NAME** = "LatLonMin"
- static const string **m_LATLONSTEP_NAME** = "LatLonStep"
- static const string **m_LON_DIM_NAME** = "Longitude"

Protected Member Functions

- virtual void **AddBratIndexData** ()
- virtual void **AddVar** (int32_t NetcdfId, const string &Name, const string &Description, const string &Unit, int32_t type=NC_NAT, const **CUIntArray** *dimValues=NULL, const **CStringArray** *dimNames=NULL, const **CIntArray** *dimIds=NULL, const **CStringMap** *mapAttributes=NULL)
- virtual void **AddVar** (const string &Name)
- virtual void **AddVar** (int32_t netcdfId, const string &name, const string &description, const string &unit, int32_t type, uint32_t dimValue, const string dimName, int32_t dimId, const **CStringMap** *mapAttributes=NULL)
- void **AddVirtualVariables** ()
- void **CheckNetCDFDimensions** ()
- virtual void **CheckVariables** ()
- uint32_t **CurrentMeasure** () const
- virtual void **FreeResources** ()
- virtual void **GetLatitudes** (double Min, double Step, uint32_t Count, double *Vector)
- virtual void **GetLongitudes** (double Min, double Step, uint32_t Count, double *Vector)
- void **Init** ()
- virtual void **LoadStructure** ()
- virtual void **SubstituteDimNames** (**CStringArray** &dimNames)

Protected Attributes

- **CNetCDFDimension** * **m_depthDim**
- uint32_t **m_depthIndex**
- **CNetCDFDimension** * **m_latDim**
- uint32_t **m_latIndex**
- **CNetCDFDimension** * **m_lonDim**
- uint32_t **m_lonIndex**
- uint32_t **m_nbDepths**
- uint32_t **m_nbLatitudes**
- uint32_t **m_nbLongitudes**

6.44.1 Detailed Description

External files access.

Version

1.0

6.44.2 Member Function Documentation

6.44.2.1 void brathl::CExternalFilesAvisoGrid::LoadStructure () [protected],[virtual]

Array of the global dimension's index

Implements **brathl::CExternalFilesNetCDF** (p. 208).

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

- ExternalFilesAvisoGrid.h
- ExternalFilesAvisoGrid.cpp

6.45 brathl::CExternalFilesJason2 Class Reference

```
#include <ExternalFilesJason2.h>
```

Inherits brathl::CExternalFilesNetCDFCF.

Inherited by brathl::CExternalFilesJason2GDR, brathl::CExternalFilesJason2SGDR, and brathl::CExternalFilesJason2SSHA.

Public Member Functions

- **CExternalFilesJason2** (const string &name="")

Static Public Member Functions

- static string **TypeOf** ()

Static Public Attributes

- static const string **m_missionName** = **CTools::StringToUpper(CMission::m_nameJ2)**

Additional Inherited Members

6.45.1 Detailed Description

Jason-2 files access.

Version

1.0

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

- ExternalFilesJason2.h
- ExternalFilesJason2.cpp

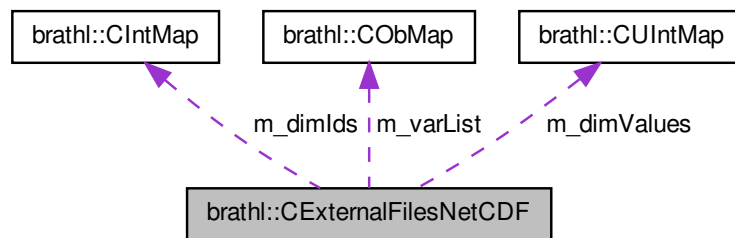
6.46 brathl::CExternalFilesNetCDF Class Reference

```
#include <ExternalFilesNetCDF.h>
```

Inherits brathl::CExternalFiles.

Inherited by brathl::CExternalFilesNetCDFCF.

Collaboration diagram for brathl::CExternalFilesNetCDF:



Public Member Functions

- virtual void **AddAttributesAsField** (CFieldNetCdf *field=NULL)
- virtual void **AddOffset** (double value, bool force=false)
- **CExternalFilesNetCDF** (const string &Name="")
- virtual void **Close** ()
- void **ExecuteExpression** (CExpression &expr, CExpressionValue &exprValue, const string &wantedUnit, CProduct *product=NULL)
- virtual CFieldNetCdf * **FindCycleField** ()
- virtual CFieldNetCdf * **FindLatField** ()
- virtual CFieldNetCdf * **FindLonField** ()
- virtual CFieldNetCdf * **FindPassField** ()
- virtual CFieldNetCdf * **FindTimeField** ()
- virtual void **GetAllValues** (const string &name, CExpressionValue &value, const string &wantedUnit)
- virtual void **GetAllValues** (const string &name, CDoubleArray &vect, const string &wantedUnit)
- virtual void **GetAllValues** (CFieldNetCdf *field, CExpressionValue &value, const string &wantedUnit)
- virtual void **GetAllValues** (CFieldNetCdf *field, const string &wantedUnit)
- int **GetAttribute** (const string &varName, const string &attName, double &attValue, bool mustExist=true, double defaultValue=CTools::m_defaultValueDOUBLE)
- int **GetAttribute** (const string &varName, const string &attName, string &attValue, bool mustExist=true, string defaultValue="")
- nc_type **GetAttributeType** (const string &attName)
- nc_type **GetAttributeType** (const string &varName, const string &attName)
- virtual void **GetDimensions** (const string &varName, CUIntArray &dimensions)
- virtual void **GetDimensions** (const string &varName, CStringArray &dimensions)
- CIntMap & **GetDimIds** ()
- CUIntMap & **GetDimValues** ()
- virtual void **GetFieldNames** (CStringArray &names)
- CFieldNetCdf * **GetFieldNetCdf** (const string &name, bool withExcept=true)
- virtual CObMap * **GetFields** ()
- CNetCDFFiles * **GetFile** ()
- int **GetGlobalAttribute** (const string &attName, double &attValue, bool mustExist=true, double defaultValue=CTools::m_defaultValueDOUBLE)
- int **GetGlobalAttribute** (const string &attName, string &attValue, bool mustExist=true, string defaultValue="")
- void **GetGlobalAttributes** (CStringMap &mapAttributes)
- void **GetGlobalAttributes** (CDoubleMap &mapAttributes)
- void **GetGlobalAttributes** (string &attributes)
- virtual string **GetName** () const

- `int32_t GetNetCdfId` (const string &name, bool withExcept=true)
- void `GetOrderedDimNames` (const string &value, CStringArray &commonDimensionNames)
- void `GetOrderedDimNames` (const CExpression &value, CStringArray &commonDimensionNames)
- void `GetOrderedDimNames` (const CStringArray *fieldNames, CStringArray &commonDimensionNames)
- void `GetOrderedDimNamesFromFieldNetcdf` (const CStringArray *fieldNames, CStringArray &commonDimensionNames)
- virtual void `GetValue` (const string &name, CExpressionValue &value, const string &wantedUnit)
- virtual void `GetValue` (const string &name, double &value, const string &wantedUnit)
- virtual void `GetValues` (const string &name, CExpressionValue &value, const string &wantedUnit)
- virtual void `GetValues` (CFieldNetCdf *field, CExpressionValue &value, const string &wantedUnit)
- `CFieldNetCdf * GetVarByAttribute` (const string &attrName, const string &attrValueToSearch)
- virtual void `GetVariables` (CStringArray &varNames)
- `nc_type GetVarType` (const string &name)
- virtual string `GetVarTypeName` (const string &name)
- virtual bool `IsAxisVar` (const string &name)
- bool `IsLatField` (CFieldNetCdf *field)
- bool `IsLonField` (CFieldNetCdf *field)
- virtual bool `IsOpened` () const
- virtual `int32_t NumberOfRecords` ()
- virtual void `Open` ()
- virtual void `SetMode` (brathl_FileMode mode)
- virtual void `SetName` (const string &Name)
- virtual void `SetOffset` (double value, bool force=false)
- virtual bool `VarExists` (const string &name)

Static Public Member Functions

- static string `TypeOf` ()

Protected Member Functions

- virtual void `AddBratIndexData` ()
- virtual void `AddVar` (int32_t NetcdfId, const string &Name, const string &Description, const string &Unit, int32_t type=NC_NAT, const CUIntArray *dimValues=NULL, const CStringArray *dimNames=NULL, const CIntArray *dimIds=NULL, const CStringMap *mapAttributes=NULL)
- virtual void `AddVar` (int32_t netcdfId, const string &name, const string &description, const string &unit, int32_t type, uint32_t dimValue, const string dimName, int32_t dimId, const CStringMap *mapAttributes=NULL)
- virtual void `AddVar` (const string &Name)
- virtual void `CheckDimensions` ()
- virtual void `CheckVariables` ()
- virtual void `FreeResources` ()
- virtual void `LoadStructure` ()=0
- void `SetOffset` (bool force=false)
- virtual void `SubstituteDimNames` (CStringArray &dimNames)

Protected Attributes

- CIntArray `m_dimIds`
- CUIntMap `m_dimValues`
- CNetCDFFiles `m_file`
- uint32_t `m_nbMeasures`
- COBMap `m_varList`

Additional Inherited Members

6.46.1 Detailed Description

External NetCdf files access.

Version

1.0

6.46.2 Member Function Documentation

6.46.2.1 `virtual void brathl::CExternalFilesNetCDF::LoadStructure () [protected], [pure virtual]`

Array of the global dimension's index

Implemented in **brathl::CExternalFilesAvisoGrid** (p. 205).

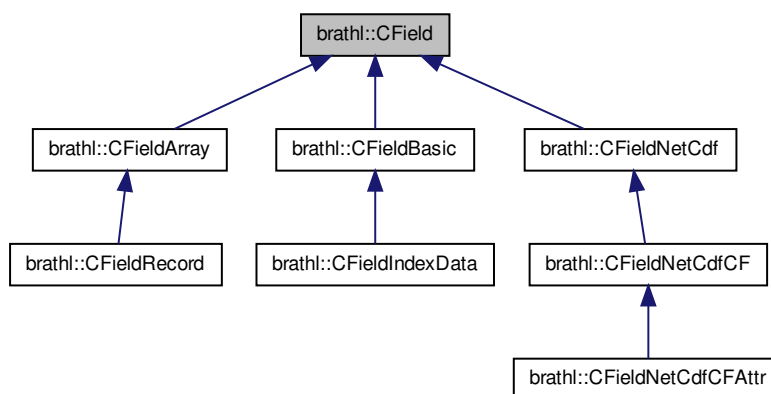
The documentation for this class was generated from the following files:

- ExternalFilesNetCDF.h
- ExternalFilesNetCDF.cpp

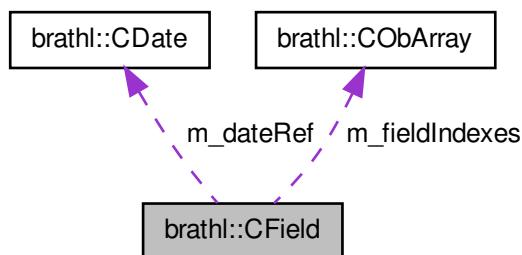
6.47 brathl::CField Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CField:



Collaboration diagram for brathl::CField:



Classes

- class **CListField**

Public Member Functions

- void **AddFieldIndexes** (CFieldIndex *value)
- void **AddFieldIndexes** (CObArray *vect, bool removeAll=true)
- virtual void **AddOffset** (double value)
- virtual void **AdjustValidMinMax** (double *data, int32_t size)
- virtual void **AdjustValidMinMax** (double value)
- **CField** ()
Ctor.
- **CField** (const string &name, const string &description="", const string &unit="")
- **CField** (CField &f)
- void **Convert** (double *data, int32_t size)
- void **ConvertDefaultValueFloat** (double *data, int32_t size)
- void **ConvertDefaultValueInt16** (double *data, int32_t size)
- void **ConvertDefaultValueInt32** (double *data, int32_t size)
- void **ConvertDefaultValueInt64** (double *data, int32_t size)
- void **ConvertDefaultValueInt8** (double *data, int32_t size)
- void **ConvertDefaultValueUInt16** (double *data, int32_t size)
- void **ConvertDefaultValueUInt32** (double *data, int32_t size)
- void **ConvertDefaultValueUInt64** (double *data, int32_t size)
- void **ConvertDefaultValueUInt8** (double *data, int32_t size)
- virtual **CFieldSet** * **CreateFieldSet** (const CField::CListField &listFields)=0
- void **DeleteFieldIndexes** ()
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **DumpFieldDictionary** (ostream &fOut=cout)
- bool **End** ()
- bool **GetConvertDate** ()
- int32_t **GetCurrentPos** ()
- coda_Cursor * **GetCursor** ()
- const CDate & **GetDateRef** ()
- const string & **GetDescription** ()

- long * **GetDim** ()
- virtual string **GetDimAsString** ()
- void **GetDimAsVector** (CUIntArray &dim)
- long **GetDimAt** (int32_t index)
- bool **GetExpandArray** ()
- CObArray * **GetFieldIndexes** ()
- virtual string **GetFullName** ()
- virtual string **GetFullNameWithRecord** ()
- virtual bool **GetHidden** ()
- virtual bool **GetHighResolution** ()
- int32_t **GetIndex** ()
- const string & **GetKey** ()
- int **GetMaxPos** ()
- const string & **GetName** ()
- coda_native_type **GetNativeType** ()
- virtual string **GetNativeTypeName** ()
- int32_t **GetNbDims** ()
- int **GetNbElts** ()
- virtual uint32_t **GetNumHighResolutionMeasure** ()
- double **GetOffset** ()
- virtual uint32_t **GetOffsetDim** ()
- virtual string **GetRecordName** ()
- coda_special_type **GetSpecialType** ()
- virtual string **GetSpecialTypeName** ()
- coda_type_class **GetTypeClass** ()
- int32_t **GetUnion** ()
- const string & **GetUnit** ()
- double **GetValidMax** ()
- double **GetValidMin** ()
- virtual int32_t **GetVirtualNbDims** ()
- void **HandleBratError** (const string &str="", int32_t errClass=BRATHL_LOGIC_ERROR)
- bool **HasDim** ()
- bool **HasEqualDims** (CField *field)
- virtual bool **HasVirtualNbDims** ()
- bool **HasXDim** ()
- bool **HasYDim** ()
- virtual bool **IsDimTransposed** ()
- bool **IsExpandArray** ()
- bool **IsFieldHasDefaultValue** ()
- bool **IsFieldNetCdfCAttr** ()
- bool **IsFixedSize** ()
- bool **IsGoToAvailableUnionField** ()
- virtual bool **IsHidden** ()
- virtual bool **IsHighResolution** ()
- bool **IsMetaData** ()
- virtual bool **IsSpecialType** ()
- bool **IsToBeRemoved** ()
- bool **IsUnion** ()
- virtual bool **IsVirtual** ()
- bool **LastRecord** ()
- const CField & **operator=** (CField &f)
- virtual void **PopCursor** ()=0
- void **PopRecordCusor** (CObList *parentFieldList)
- virtual void **PushPos** ()=0
- virtual void **Read** (CDoubleArray &vect, bool skip=false)

- virtual void **Read** (double *data, bool skip=false)
- virtual void **Read** (string &value, bool skip=false)
- virtual void **ReadParent** (CDoubleArray &vect, CFieldRecord *parentField)
- virtual void **ReadParent** (CDoubleArray &vect, CObList *parentFieldList)
- void **Set** (CField &f)
- void **SetConvertDate** (bool value)
- void **SetCurrentPos** (int32_t currentPos)
- void **SetCurrentPosToLast** ()
- void **SetCursor** (coda_Cursor &cursor)
- void **SetDateRef** (brathl_refDate refDate)
- void **SetDateRef** (const CDate &value)
- void **SetDefaultValue** (double *data, int32_t size)
- void **SetDescription** (const string &description)
- void **SetDim** (int32_t nbDims, const long dim[])
- void **SetDim** (int32_t nbDims, const CUIntArray &dim)
- void **SetDim** (const CUIntArray &dim)
- void **SetDim** (const CUIntArray *dim)
- void **SetDim** (int32_t nbElts)
- void **SetExpandArray** (bool value)
- void **SetFieldHasDefaultValue** (bool value)
- void **SetFixedSize** (bool isFixedSize)
- void **SetGoToAvailableUnionField** (bool value)
- virtual void **SetHidden** (bool value)
- virtual void **SetHighResolution** (bool value)
- void **SetIndex** (int32_t index)
- void **SetKey** (const string &key)
- void **SetMetaData** (bool metaData)
- void **SetName** (const string &name)
- void **SetNativeType** (coda_native_type nativeType)
- virtual void **SetNumHighResolutionMeasure** (uint32_t value)
- virtual void **SetOffset** (double value)
- void **SetSpecialType** (coda_special_type specialType)
- void **SetToBeRemoved** (bool value)
- void **SetTypeClass** (coda_type_class typeClass)
- void **SetUnion** (int32_t value)
- virtual void **SetUnit** (const string &unit)
- void **SetValidMax** (double value)
- void **SetValidMin** (double value)
- virtual void **SetVirtual** (bool value)
- bool **TransposeDim** ()
- bool **TransposeValues** (double *data, int32_t size)
- bool **UnitIsDate** ()
- virtual ~CField ()

Dtor.

Static Public Member Functions

- static void **AdjustValidMinMax** (double *data, int32_t size, double &min, double &max)
- static void **AdjustValidMinMax** (double value, double &min, double &max)
- static CFieldNetCdfCFAttr * **GetFieldNetCdfCFAttr** (CBratObject *ob, bool withExcept=true)
- static CFieldNetCdfIndexData * **GetFieldNetCdfIndexData** (CBratObject *ob, bool withExcept=true)
- static bool **IsFieldNetCdfCFAttr** (CBratObject *ob)

Static Public Attributes

- static const string **m_BRAT_INDEX_DATA_DESC** = "data index"
- static const string **m_BRAT_INDEX_DATA_NAME** = "brat_index_data"

Protected Member Functions

- void **Init** ()

Protected Attributes

- bool **m_convertDate**
- int32_t **m_currentPos**
- coda_Cursor **m_cursor**
- CDate **m_dateRef**
- string **m_description**
- long **m_dim** [MAX_NUM_DIMS]
- bool **m_dimsTransposed**
- bool **m_expandArray**
- bool **m_fieldHasDefaultValue**
- CObArray * **m_fieldIndexes**
- string **m_fullName**
- bool **m_goToAvailableUnionField**
- bool **m_hidden**
- bool **m_highResolution**
- int32_t **m_index**
- bool **m_isFixedSize**
- int32_t **m_isUnion**
- string **m_key**
- int32_t **m_maxPos**
- bool **m_metadata**
- string **m_name**
- coda_native_type **m_nativeType**
- int32_t **m_nbDims**
- uint32_t **m_numHighResolutionMeasure**
- double **m_offset**
- string **m_recordName**
- coda_special_type **m_specialType**
- bool **m_toBeRemoved**
- coda_type_class **m_typeClass**
- string **m_unit**
- bool **m_unitIsDate**
- double **m_validMax**
- double **m_validMin**
- bool **m_virtualField**

6.47.1 Detailed Description

Field management base classe.

Version

1.0

6.47.2 Member Data Documentation

6.47.2.1 long brathl::CField::m_dim[MAX_NUM_DIMS] [protected]

total number of dimensions

6.47.2.2 bool brathl::CField::m_isFixedSize [protected]

(maximum) dimensions

6.47.2.3 double brathl::CField::m_validMax [protected]

Valid max value

6.47.2.4 double brathl::CField::m_validMin [protected]

Valid min value

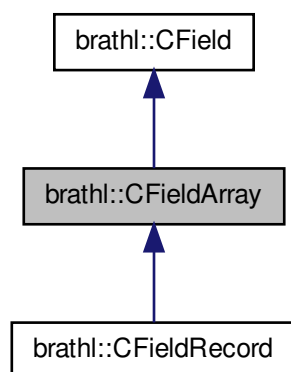
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

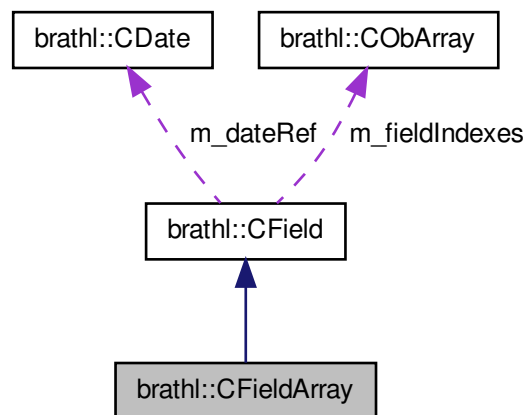
6.48 brathl::CFieldArray Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CFieldArray:



Collaboration diagram for brathl::CFieldArray:



Public Member Functions

- **CFieldArray** ()
Ctor.
- **CFieldArray** (const string &name, const string &description="", const string &unit="")
- **CFieldArray** (int32_t nbDims, const long dim[], const string &name, const string &description="", const string &unit="")
- **CFieldArray** (CFieldArray &f)
- void **CreateFieldIndexes** (CObArray &vect)
- virtual **CFieldSet** * **CreateFieldSet** (const CField::CListField &listFields)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **DumpFieldDictionary** (ostream &fOut=cout)
- virtual uint32_t **GetOffsetDim** ()
- virtual int32_t **GetVirtualNbDims** ()
- const **CFieldArray** & **operator=** (CFieldArray &f)
- virtual void **PopCursor** ()
- virtual void **PushPos** ()
- virtual void **PushPos** (int32_t iDim)
- virtual void **Read** (CDoubleArray &vect, bool skip=false)
- virtual void **Read** (double *data, bool skip=false)
- void **Set** (CFieldArray &f)
- virtual ~**CFieldArray** ()
Dtor.

Additional Inherited Members

6.48.1 Detailed Description

Field of type 'array" management classes.

Version

1.0

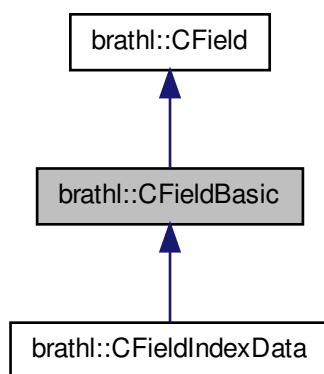
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

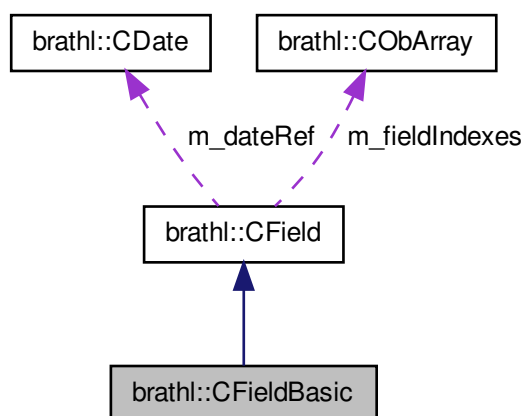
6.49 brathl::CFieldBasic Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CFieldBasic:



Collaboration diagram for brathl::CFieldBasic:



Public Member Functions

- **CFieldBasic** ()
Ctor.
- **CFieldBasic** (long length, const string &name, const string &description, const string &unit)
- **CFieldBasic** (CFieldBasic &f)
- virtual **CFieldSet** * **CreateFieldSet** (const CField::CListField &listFields)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **DumpFieldDictionary** (ostream &fOut=cout)
- const **CFieldBasic** & **operator=** (CFieldBasic &f)
- virtual void **PopCursor** ()
- virtual void **PushPos** ()
- virtual void **Read** (CDoubleArray &vect, bool skip=false)
- virtual void **Read** (double *data, bool skip=false)
- virtual void **Read** (string &data, bool skip=false)
- void **Set** (CFieldBasic &f)
- virtual **~CFieldBasic** ()
Dtor.

Public Attributes

- long **m_length**

Additional Inherited Members

6.49.1 Detailed Description

Field of type 'basic' management classes.

Version

1.0

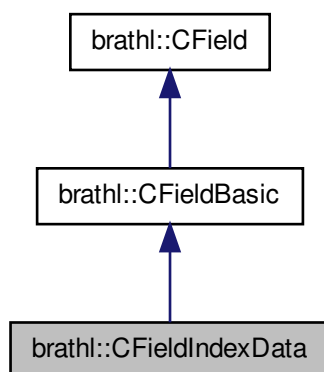
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

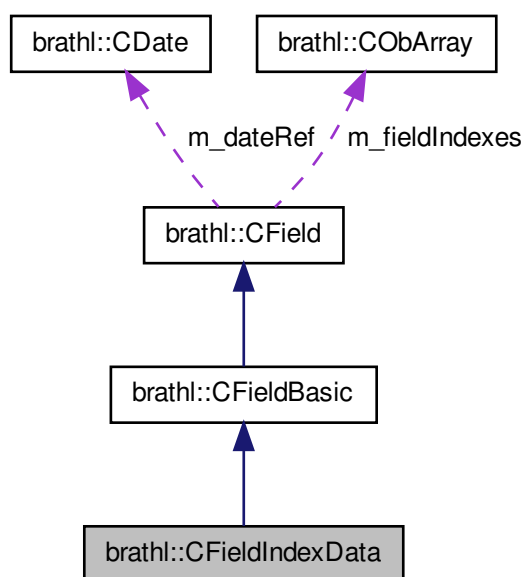
6.50 brathl::CFieldIndexData Class Reference

```
#include <Field.h>
```

Inheritance diagram for bratl::CFieldIndexData:



Collaboration diagram for bratl::CFieldIndexData:



Public Member Functions

- **CFieldIndexData** ()
Ctor.
- **CFieldIndexData** (const string &name, const string &description, const string &unit="")
- **CFieldIndexData** (CFieldIndexData &f)

- virtual **CFieldSet** * **CreateFieldSet** (const **CField::CListField** &listFields)
- virtual void **Dump** (ostream &fOut=cerr)

Dump function.

- virtual void **DumpFieldDictionary** (ostream &fOut=cout)
- double **GetValue** ()
- const **CFieldIndexData** & **operator=** (**CFieldIndexData** &f)
- virtual void **PopCursor** ()
- virtual void **PushPos** ()
- virtual void **Read** (**CDoubleArray** &vect, bool skip=false)
- virtual void **Read** (double *data, bool skip=false)
- virtual void **Read** (string &data, bool skip=false)
- virtual void **Read** (double &value)
- virtual double **Read** ()
- void **Set** (**CFieldIndexData** &f)
- virtual ~**CFieldIndexData** ()

Dtor.

Protected Member Functions

- void **Init** ()

Additional Inherited Members

6.50.1 Detailed Description

Field of type 'basic' management classes.

Version

1.0

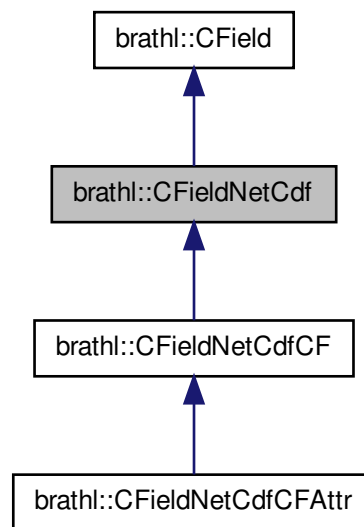
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

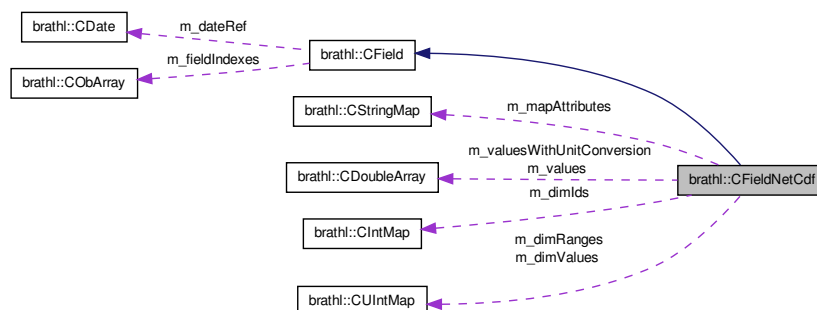
6.51 brathl::CFieldNetCdf Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CFieldNetCdf:



Collaboration diagram for brathl::CFieldNetCdf:



Public Member Functions

- void **AdjustValidMinMaxFromValues** ()
- **CFieldNetCdf** ()
Ctor.
- **CFieldNetCdf** (const string &name, const string &description="", const string &unit="", int32_t netCdfId=NC_GLOBAL, int32_t type=NC_NAT, const **CUIntArray** *dimValues=NULL, const CStringArray *dimNames=NULL, const **CIntArray** *dimIds=NULL, const **CDoubleArray** *values=NULL)
- **CFieldNetCdf** (**CFieldNetCdf** &f)
- virtual CBratObject * **Clone** ()
- **CFieldNetCdf** * **CloneThis** ()
- virtual **CFieldSet** * **CreateFieldSet** (const **CField::CListField** &listFields)

- virtual **CFieldSet** * **CreateFieldSet** ()
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **DumpFieldDictionary** (ostream &fOut=cout)
- void **EmptyValues** ()
- double **GetAddOffset** ()
- virtual string **GetAttribute** (const string attrName)
- const **CStringMap** & **GetAttributes** ()
- int32_t **GetCounFromDimCountArray** ()
- const **CIntMap** & **GetDimIds** ()
- void **GetDimIdsAsArray** (**CIntArray** &values, bool bRemoveAll=true)
- const **CStringArray** & **GetDimNames** ()
- uint32_t **GetDimRange** (const string &dimName)
- const **CUIntMap** & **GetDimRanges** ()
- uint32_t * **GetDimsCountArray** ()
- uint32_t * **GetDimsIndexArray** ()
- const **CUIntMap** & **GetDimValues** ()
- void **GetDimValuesAsArray** (**CUIntArray** &values, bool bRemoveAll=true)
- double **GetFillValue** ()
- virtual string **GetFullName** ()
- virtual string **GetFullNameWithRecord** ()
- virtual string **GetMostExplicitName** ()
- int32_t **GetNativeType** ()
- virtual string **GetNativeTypeName** ()
- int32_t **GetNetCdfId** ()
- **CUnit** * **GetNetCdfUnit** ()
- int32_t **GetPosFromDimIndexArray** ()
- virtual string **GetRecordName** ()
- double **GetScaleFactor** ()
- int32_t **GetSpecialType** ()
- virtual string **GetSpecialTypeName** ()
- int32_t **GetType** ()
- virtual string **GetTypeName** ()
- virtual **CDoubleArray** & **GetValues** ()
- double * **GetValuesAsArray** ()
- virtual **CDoubleArray** & **GetValuesWithUnitConversion** (const string &wantedUnit)
- virtual int32_t **GetVirtualNbDims** ()
- virtual void **InitDimIndexes** (uint32_t value)
- virtual void **InitDimsIndexToMax** ()
- virtual void **InitDimsIndexToMax** (uint32_t index)
- bool **IsAtBeginning** ()
- virtual bool **IsSpecialType** ()
- uint32_t * **NewDimIndexArray** (**CFieldNetCdf** *fromField=NULL)
- bool **NextIndex** ()
- const **CFieldNetCdf** & **operator=** (**CFieldNetCdf** &f)
- virtual void **PopCursor** ()
- bool **PrevIndex** ()
- virtual void **PushPos** ()
- virtual void **Read** (**CDoubleArray** &vect, bool skip=false)
- virtual void **Read** (**CExpressionValue** &value, bool skip=false)
- **NetCDFVarKind** **SearchDimKind** ()
- void **Set** (**CFieldNetCdf** &f)
- void **SetAddOffset** (double value)
- void **SetAtBeginning** (bool value)

- virtual void **SetAttributes** (const **CStringMap** &mapAttributes)
- virtual void **SetAttributes** (const **CStringMap** *mapAttributes)
- void **SetDimIds** (const **CIntMap** &dimIds)
- void **SetDimIds** (const **CIntMap** *dimIds)
- virtual void **SetDimInfo** (const CStringArray &dimNames, const **CIntArray** &dimIds, const **CUIntArray** &dimValues)
- virtual void **SetDimInfo** (const CStringArray *dimNames, const **CIntArray** *dimIds, const **CUIntArray** *dimValues)
- virtual void **SetDimNames** (const CStringArray &dimNames)
- virtual void **SetDimNames** (const CStringArray *dimNames)
- virtual void **SetDimValues** (const **CUIntMap** &dimValues)
- virtual void **SetDimValues** (const **CUIntMap** *dimValues)
- void **SetFillValue** (double value)
- virtual void **SetIndex** (const string &dimName, uint32_t index, uint32_t count)
- void **SetNativeType** (int32_t type)
- void **SetNetCdfId** (int32_t id)
- void **SetScaleFactor** (double value)
- virtual void **SetType** (int32_t type)
- virtual void **SetUnit** (const string &unit)
- virtual void **SetUnit** (const CUnit &unit)
- virtual void **SetValues** (double values)
- virtual void **SetValues** (double *values, size_t length)
- virtual void **SetValues** (const **CDoubleArray** &values)
- virtual void **SetValues** (const **CDoubleArray** *values)
- virtual void **SetValues** (const **CInt16Array** &values)
- virtual void **SetValues** (const **CInt16Array** *values)
- virtual void **SetValues** (const **CInt8Array** &values)
- virtual void **SetValues** (const **CInt8Array** *values)
- virtual void **SetValues** (const **CIntArray** &values)
- virtual void **SetValues** (const **CIntArray** *values)
- virtual void **SetValues** (const **CUInt8Array** &values)
- virtual void **SetValues** (const **CUInt8Array** *values)
- virtual void **SetValues** (const **CFloatArray** &values)
- virtual void **SetValues** (const **CFloatArray** *values)
- virtual void **SetValues** (const string &values)
- void **SetValuesAsArray** ()
- void **SetValuesAsArray** (const **CDoubleArray** &values)
- void **SetValuesAsArray** (const **CDoubleArray** *values)
- virtual ~**CFieldNetCdf** ()

Dtor.

Protected Member Functions

- void **DeleteDimIndexArray** ()
- void **DeleteValuesAsArray** ()
- void **Init** ()

Protected Attributes

- double **m_addOffset**
- bool **m_atBeginning**
- **CIntMap** **m_dimIds**
- CStringArray **m_dimNames**
- **CUIntMap** **m_dimRanges**
- uint32_t * **m_dimsCountArray**
- uint32_t * **m_dimsIndexArray**
- **CUIntMap** **m_dimValues**
- double **m_fillValue**
- **CStringMap** **m_mapAttributes**
- int32_t **m_netCdfId**
- CUnit **m_netCdfUnit**
- double **m_scaleFactor**
- int32_t **m_type**
- **CDoubleArray** **m_values**
- double * **m_valuesAsArray**
- **CDoubleArray** **m_valuesWithUnitConversion**

Additional Inherited Members

6.51.1 Detailed Description

Field from a NetCdf file management classes.

Version

1.0

6.51.2 Member Data Documentation

6.51.2.1 double brathl::CFieldNetCdf::m_addOffset [protected]

data add offset

Referenced by Dump().

6.51.2.2 bool brathl::CFieldNetCdf::m_atBeginning [protected]

'At beginning" flag

Referenced by Dump().

6.51.2.3 CIntMap brathl::CFieldNetCdf::m_dimIds [protected]

Map of the dimension's ids of the field (key is dim. name)

Referenced by Dump().

6.51.2.4 CStringArray brathl::CFieldNetCdf::m_dimNames [protected]

Array of the dimension's names of the field (index is dim. range)

Referenced by Dump().

6.51.2.5 CUIntMap brathl::CFieldNetCdf::m_dimRanges [protected]

Map of the dimension's range of the field (key is dim. name)

Referenced by Dump().

6.51.2.6 `uint32_t* bratl::CFieldNetCdf::m_dimsCountArray` [protected]

Array of the dimension count for reading

Referenced by Dump().

6.51.2.7 `uint32_t* bratl::CFieldNetCdf::m_dimsIndexArray` [protected]

Array of the dimension's index

Referenced by Dump().

6.51.2.8 `CUIntMap bratl::CFieldNetCdf::m_dimValues` [protected]

Map of the dimension's values of the field (key is dim. name)

Referenced by Dump().

6.51.2.9 `double bratl::CFieldNetCdf::m_fillValue` [protected]

data default value (fill value)

Referenced by Dump().

6.51.2.10 `CStringMap bratl::CFieldNetCdf::m_mapAttributes` [protected]

Map of the netcdf attributes (as string representation).

Referenced by Dump().

6.51.2.11 `int32_t bratl::CFieldNetCdf::m_netCdfId` [protected]

The netcdf external id

Referenced by Dump().

6.51.2.12 `CUnit bratl::CFieldNetCdf::m_netCdfUnit` [protected]

The netcdf unit

Referenced by Dump().

6.51.2.13 `double bratl::CFieldNetCdf::m_scaleFactor` [protected]

data scale factor

Referenced by Dump().

6.51.2.14 `int32_t bratl::CFieldNetCdf::m_type` [protected]

The netcdf external data types

Referenced by Dump().

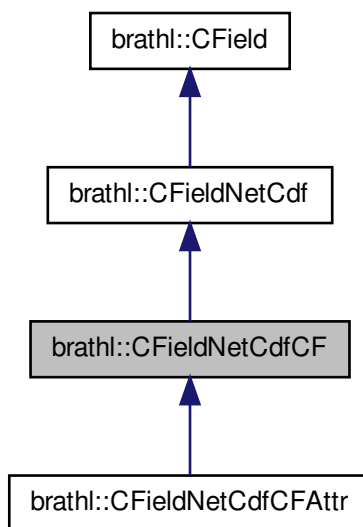
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

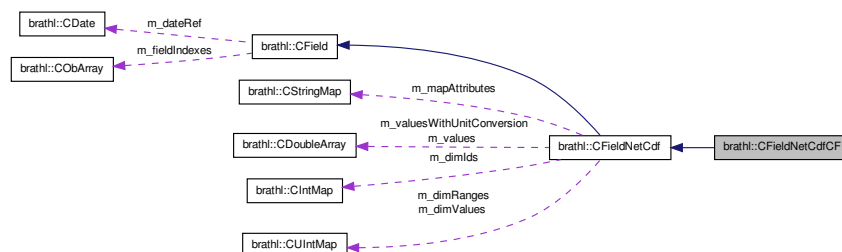
6.52 bratl::CFieldNetCdfCF Class Reference

```
#include <Field.h>
```

Inheritance diagram for bratl::CFieldNetCdfCF:



Collaboration diagram for bratl::CFieldNetCdfCF:



Public Member Functions

- **CFieldNetCdfCF** ()

Ctor.

- **CFieldNetCdfCF** (const string &name, const string &description="", const string &unit="", int32_t netCdfId=NC_GLOBAL, int32_t type=NC_NAT, const **CUIntArray** *dimValues=NULL, const CStringArray *dimNames=NULL, const **CIntArray** *dimIds=NULL, const **CDoubleArray** *values=NULL)
- **CFieldNetCdfCF** (**CFieldNetCdfCF** &f)
- virtual **CBratObject** * **Clone** ()
- virtual void **Dump** (ostream &fOut=cerr)

Dump function.

- virtual void **DumpFieldDictionary** (ostream &fOut=cout)
- virtual string **GetDimAsString** ()
- string **GetDimAsStringWithIndexes** ()
- string **GetDimAsStringWithNames** ()

- const **CFieldNetCdfCF** & **operator=** (**CFieldNetCdfCF** &f)
 - void **Set** (**CFieldNetCdfCF** &f)
 - virtual ~**CFieldNetCdfCF** ()
- Dtor.*

Protected Member Functions

- void **Init** ()

Additional Inherited Members

6.52.1 Detailed Description

Field from a NetCdf file management classes.

Version

1.0

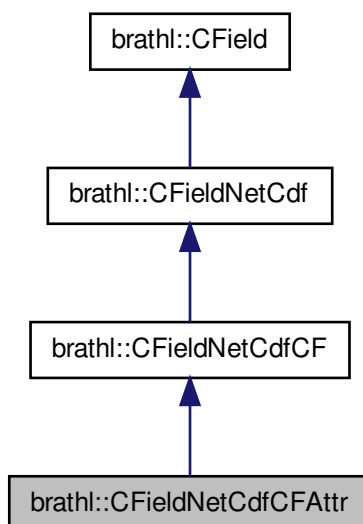
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

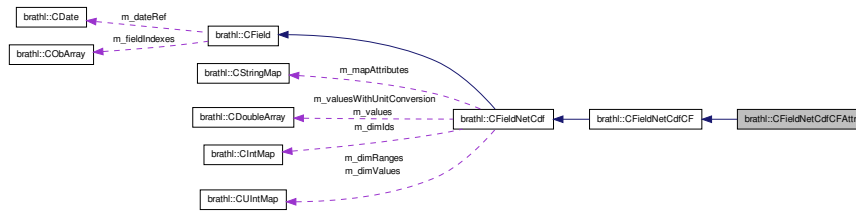
6.53 brathl::CFieldNetCdfCFAttr Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CFieldNetCdfCFAttr:



Collaboration diagram for brathl::CFieldNetCdfCFAttr:



Public Member Functions

- **CFieldNetCdfCFAttr** ()

Ctor.

- **CFieldNetCdfCFAttr** (CNetCDFVarDef *netCDFVarDef, CNetCDFAttr *netCDFAttr)
- **CFieldNetCdfCFAttr** (CNetCDFAttr *netCDFAttr)
- **CFieldNetCdfCFAttr** (CFieldNetCdfCFAttr &f)
- virtual CBratObject * **Clone** ()
- **CFieldNetCdfCFAttr** * **CloneThis** ()
- void **DeleteNetCDFAttr** ()
- virtual void **Dump** (ostream &fOut=cerr)
- virtual void **DumpFieldDictionary** (ostream &fOut=cout)
- virtual string **GetMostExplicitName** ()
- CNetCDFAttr * **GetNetCDFAttr** ()
- const string & **GetRelatedVarName** ()
- bool **IsFieldNetCdfCFAttrGlobal** ()
- bool **IsFieldNetCdfCFAttrVariable** ()
- const **CFieldNetCdfCFAttr** & **operator=** (**CFieldNetCdfCFAttr** &f)
- void **Set** (**CFieldNetCdfCFAttr** &f)
- virtual void **SetAttributes** (const **CStringMap** &mapAttributes)
- virtual void **SetAttributes** (const **CStringMap** *mapAttributes)
- void **SetInfoFromAttr** (CNetCDFVarDef *netCDFVarDef=NULL)
- void **SetInfoFromAttr** (CNetCDFAttr *netCDFAttr, CNetCDFVarDef *netCDFVarDef=NULL)
- void **SetNetCDFAttr** (CNetCDFAttr *value)
- void **SetRelatedVarName** (const string &value)
- virtual void **SetType** (int32_t type)
- void **SetValuesFromAttr** ()
- void **SetValuesFromAttr** (CNetCDFAttr *netCDFAttr)
- virtual ~**CFieldNetCdfCFAttr** ()

Dtor.

Static Public Member Functions

- static bool **IsFieldNetCdfCFAttrGlobal** (CBratObject *ob)
- static bool **IsFieldNetCdfCFAttrVariable** (CBratObject *ob)

Protected Member Functions

- void **Init** ()

Protected Attributes

- CNetCDFAttr * **m_netCDFAttr**
- string **m_relatedVarName**

Additional Inherited Members

6.53.1 Detailed Description

Field from a NetCdf Attribute file management classes.

Version

1.0

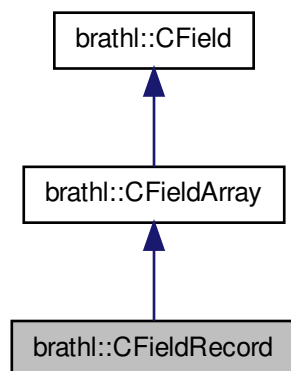
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

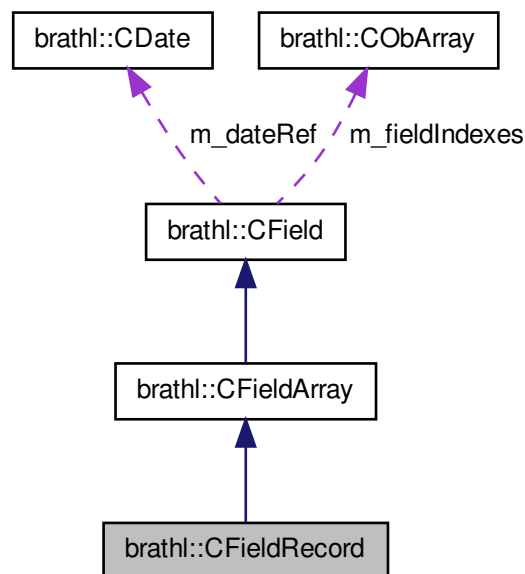
6.54 brathl::CFieldRecord Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CFieldRecord:



Collaboration diagram for brathl::CFieldRecord:



Public Member Functions

- **CFieldRecord** ()
Ctor.
- **CFieldRecord** (int32_t nbFields, const string &name, const string &description="", const string &unit="")
- **CFieldRecord** (int32_t nbDims, const long dim[], int32_t nbFields, const string &name, const string &description="", const string &unit="")
- **CFieldRecord** (CFieldRecord &f)
- virtual **CFieldSet** * **CreateFieldSet** (const CField::CListField &listFields)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **DumpFieldDictionary** (ostream &fOut=cout)
- int32_t **GetNbFields** ()
- virtual int32_t **GetVirtualNbDims** ()
- const **CFieldRecord** & **operator=** (CFieldRecord &f)
- virtual void **PopCursor** ()
- virtual void **PushPos** ()
- virtual void **PushPos** (int32_t iDim)
- virtual void **Read** (CDoubleArray &vect, bool skip=false)
- virtual void **Read** (double *data, bool skip=false)
- void **Set** (CFieldRecord &f)
- void **SetNbFields** (int32_t value)
- virtual ~**CFieldRecord** ()
Dtor.

Protected Attributes

- int32_t **m_nbFields**

Additional Inherited Members

6.54.1 Detailed Description

Field of type 'record' management classes.

Version

1.0

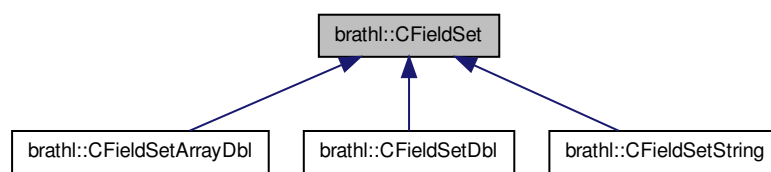
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

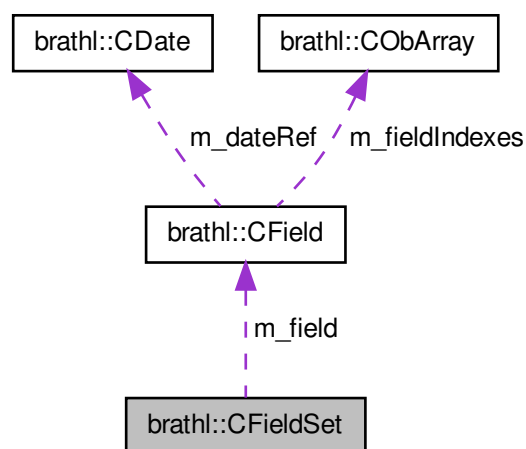
6.55 bratl::CFieldSet Class Reference

```
#include <Field.h>
```

Inheritance diagram for bratl::CFieldSet:



Collaboration diagram for bratl::CFieldSet:



Public Member Functions

- **CFieldSet** (const string &name="")
Ctor.
- **CFieldSet** (**CFieldSet** &f)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual **CField** * **GetField** ()
- const string & **GetName** ()
- virtual void **Insert** (const **CDoubleArray** &vect, bool bRemove=false)=0
- virtual void **Insert** (double value, bool bRemove=false)=0
- virtual void **Insert** (const string &value, bool bRemove=false)=0
- **CFieldSet** & **operator=** (**CFieldSet** &o)
- virtual void **SetField** (**CField** *value)
- virtual ~**CFieldSet** ()
Dtor.

Protected Member Functions

- void **Copy** (**CFieldSet** &f)

Protected Attributes

- **CField** * **m_field**
- string **m_name**

6.55.1 Detailed Description

a base class for set of field value.

Version

1.0

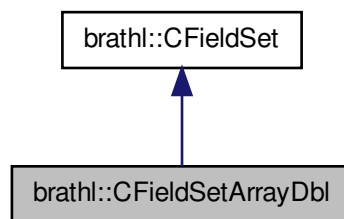
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

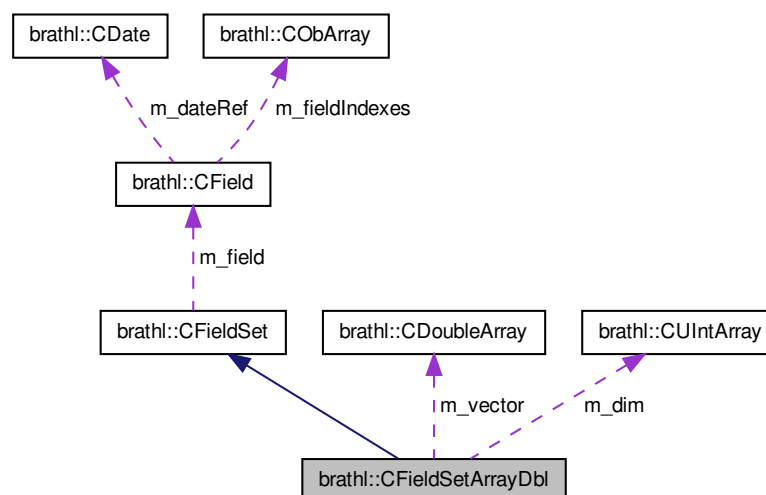
6.56 brathl::CFieldSetArrayDbI Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CFieldSetArrayDbI:



Collaboration diagram for brathl::CFieldSetArrayDbI:



Public Member Functions

- **CFieldSetArrayDbI** (const string &name="")
Ctor.
- **CFieldSetArrayDbI** (CFieldSetArrayDbI &f)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- **CDoubleArray** & **GetDataVector** ()
- virtual void **Insert** (const **CDoubleArray** &vect, bool bRemove=false)
- virtual void **Insert** (double value, bool bRemove=false)
- virtual void **Insert** (const string &value, bool bRemove=false)
- **CFieldSetArrayDbI** & **operator=** (CFieldSetArrayDbI &o)
- virtual ~**CFieldSetArrayDbI** ()
Dtor.

Public Attributes

- **CIntArray** m_dim
- int32_t m_nbDims
- **CDoubleArray** m_vector

Protected Member Functions

- void **Copy** (CFieldSetArrayDbI &f)

Additional Inherited Members

6.56.1 Detailed Description

a set of double array field value.

Version

1.0

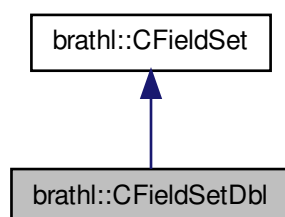
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

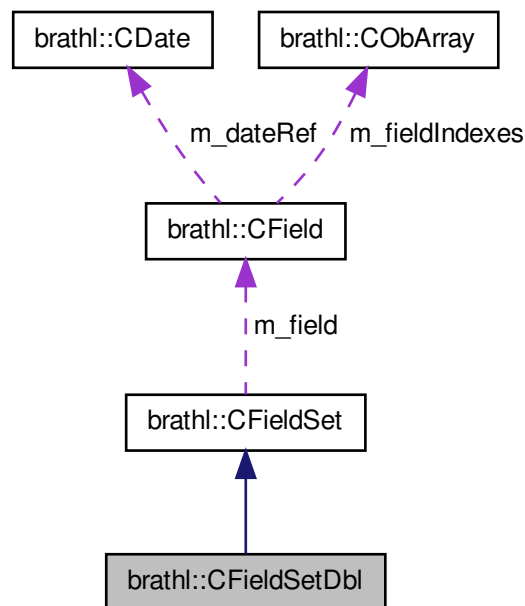
6.57 bratl::CFieldSetDbI Class Reference

```
#include <Field.h>
```

Inheritance diagram for bratl::CFieldSetDbI:



Collaboration diagram for brathl::CFieldSetDbI:



Public Member Functions

- `int32_t AsInt32 ()`
- `int32_t AsUInt32 ()`
- `CFieldSetDbI (const string &name="")`
Ctor.
- `CFieldSetDbI (CFieldSetDbI &f)`
- `virtual void Dump (ostream &fOut=cerr)`
Dump fonction.
- `double GetData ()`
- `double & GetDataRef ()`
- `virtual void Insert (const CDoubleArray &vect, bool bRemove=false)`
- `virtual void Insert (double value, bool bRemove=false)`
- `virtual void Insert (const string &value, bool bRemove=false)`
- `CFieldSetDbI & operator= (CFieldSetDbI &o)`
- `void SetData (double value)`
- `virtual ~CFieldSetDbI ()`
Dtor.

Public Attributes

- `double m_value`

Protected Member Functions

- `void Copy (CFieldSetDbI &f)`

Additional Inherited Members

6.57.1 Detailed Description

a set of double field value.

Version

1.0

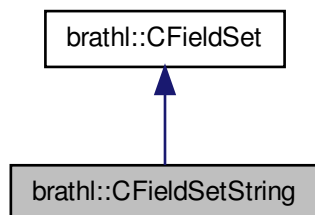
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

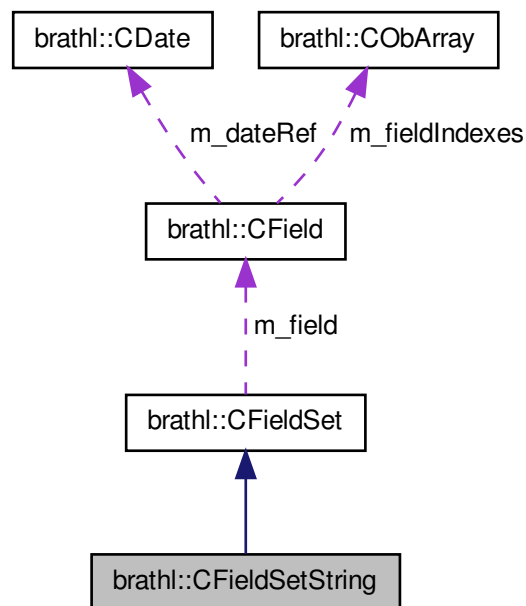
6.58 brathl::CFieldSetString Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CFieldSetString:



Collaboration diagram for brathl::CFieldSetString:



Public Member Functions

- **CFieldSetString** (const string &name="")
Ctor.
- **CFieldSetString** (CFieldSetString &f)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- string **GetData** ()
- string & **GetDataRef** ()
- virtual void **Insert** (const CDoubleArray &vect, bool bRemove=false)
- virtual void **Insert** (double value, bool bRemove=false)
- virtual void **Insert** (const string &value, bool bRemove=false)
- **CFieldSetString & operator=** (CFieldSetString &o)
- void **SetData** (const string &value)
- virtual ~**CFieldSetString** ()
Dtor.

Public Attributes

- string **m_value**

Protected Member Functions

- void **Copy** (CFieldSetString &f)

Additional Inherited Members

6.58.1 Detailed Description

a set of string field value.

Version

1.0

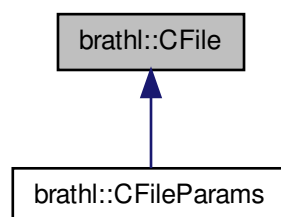
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

6.59 brathl::CFile Class Reference

```
#include <File.h>
```

Inheritance diagram for brathl::CFile:



Public Types

- enum **openFlags** {
modeRead = 0x0001, **modeWrite** = 0x0002, **modeAppend** = 0x0004, **modeReadWrite** = 0x0008,
modeRWCreate = 0x0010, **modeReadAppend** = 0x0020, **typeText** = 0x4000, **typeBinary** = static_
cast<int32_t>(0x8000) }

Public Member Functions

- **CFile** ()
*Empty **CFile** (p. 236) ctor.*
- **CFile** (const string &name, uint32_t mode=**modeRead**|**typeBinary**)
- bool **Close** ()
- bool **Delete** ()
- virtual void **Dump** (ostream &fOut=cerr)
Gets the las error message encountered.
- bool **Duplicate** (const string &newFileName)
- void **Flush** ()
- const string & **GetFileName** ()

- long **GetLength** ()
Returns the current length of the file.
- uint32_t **GetMode** ()
- long **GetPosition** ()
Returns the current position of the file pointer.
- bool **GetStatus** (struct stat &fileStatus)
- bool **IsOpen** ()
- bool **Open** (const string &name, uint32_t mode=**modeRead**|**typeBinary**)
- bool **Open** ()
- int32_t **ReadLine** (char *lineRead, uint32_t size)
- int32_t **ReadLineData** (char *lineRead, uint32_t size)
- int32_t **ReadToBuffer** (char *destinationBuffer, uint32_t numBytesToRead=CFile::m_maxBufferToRead)
- bool **Rename** (const string &newName)
- bool **SeekToBegin** ()
- bool **SeekToEnd** ()
- bool **SetBufferingMode** (bool mode=true)
- bool **SetPosition** (long positionOffset)
- bool **Write** (const int character)
- bool **Write** (const string &str)
- bool **Write** (const char *str)
- bool **WriteChar** (const int character)
- uint32_t **WriteFromBuffer** (const char *sourceBuffer, uint32_t sourceBufferLength)
- bool **WriteString** (const char *str)
- virtual ~**CFile** ()
Destructor.

Static Public Member Functions

- static bool **Delete** (const string &filename)
- static bool **GetStatus** (const string &filename, struct stat &fileStatus)
- static bool **Rename** (const string &oldName, const string &newName)

Protected Attributes

- char **m_lastError** [BRATHL_MAX_ERRMSG_LEN+1]
last error message

6.59.1 Detailed Description

File management class.

This class provides unbuffered, binary and ascii disk input/output services.

While managing the file, if an error occurred, a **CFileException** (p. 243) is raised.

Version

1.0

6.59.2 Member Enumeration Documentation

6.59.2.1 enum brathl::CFile::openFlags

File access mode enumeration: Flags can be combined by using the bitwise-OR (|) operator

Enumerator:

modeRead Opens for reading. If the file does not exist or cannot be found, open fails.

modeWrite Opens an empty file for writing. If the given file exists, its contents are destroyed.

modeAppend Opens for writing at the end of the file (appending) without removing the EOF marker before writing new data to the file; creates the file first if it doesn't exist.

modeReadWrite Opens for both reading and writing. (The file must exist.)

modeRWCreate Opens an empty file for both reading and writing. If the given file exists, its contents are destroyed.

modeReadAppend Opens for reading and appending; the appending operation includes the removal of the EOF marker before new data is written to the file and the EOF marker is restored after writing is complete; creates the file first if it doesn't exist.

typeText Open in text (translated) mode.

typeBinary Open in binary (untranslated) mode.

6.59.3 Constructor & Destructor Documentation

6.59.3.1 brathl::CFile::CFile (const string & name, uint32_t mode = modeRead|typeBinary)

Creates new **CFile** (p. 236) object and opens the file. If an error occurred, a **CFileException** (p. 243) is raised.

Parameters

<i>name</i>	[in] : full name of the file;
<i>mode</i>	[in] : access mode - default value : modeRead typeBinary (see openFlags (p. 238));

6.59.4 Member Function Documentation

6.59.4.1 bool brathl::CFile::Close ()

Closes file object. **IsOpen()** (p. 240) and **Open()** (p. 240) are the only functions available just after this operation.

Returns

true on success, otherwise false

Referenced by brathl::CFileParams::Load(), and brathl::CMission::LoadAliasName().

6.59.4.2 bool brathl::CFile::Delete ()

Closes file object and deletes (removes) the file. **IsOpen()** (p. 240) and **Open()** (p. 240) are the only functions available just after this operation.

Returns

true on success, otherwise false

6.59.4.3 bool brathl::CFile::Delete (const string & filename) [static]

Deletes (removes) a file.

Parameters

<i>filename</i>	[in] : file to delete/remove IsOpen() (p. 240) and Open() (p. 240) are the only functions available just after this operation.
-----------------	--

Returns

true on success, otherwise false

6.59.4.4 void brathl::CFile::Dump (ostream & *fOut* = cerr) [virtual]

Gets the las error message encountered.

Dump fonction

Reimplemented in **brathl::CFileParams** (p. 246).

6.59.4.5 bool brathl::CFile::Duplicate (const string & *newFileName*)

Creates a copy of current file with the newFileName. If file with specified filename exists, it's contents are erased.

Parameters

<i>newFileName</i>	[in] : copy to file name
--------------------	--------------------------

Returns

true on success, otherwise false

References IsOpen(), and WriteFromBuffer().

6.59.4.6 const string & brathl::CFile::GetFileName ()

Gets the name of the file

6.59.4.7 uint32_t brathl::CFile::GetMode ()

Gets the name of the file

6.59.4.8 bool brathl::CFile::GetStatus (struct stat & *fileStatus*)

Gets information about the file.

Parameters

<i>fileStatus</i>	[in] : structure to store results
-------------------	-----------------------------------

Returns

true on success, otherwise false

6.59.4.9 bool brathl::CFile::GetStatus (const string & *filename*, struct stat & *fileStatus*) [static]

Gets information about a file.

Parameters

<i>filename</i>	[in] : file to get the status
<i>fileStatus</i>	[in] : structure to store results

Returns

true on success, otherwise false

Referenced by Open().

6.59.4.10 bool brathl::CFile::IsOpen ()

Tests if file is opened or not

Returns

true if opened, false otherwise

Referenced by Duplicate(), brathl::CFileParams::Load(), and brathl::CMission::LoadAliasName().

6.59.4.11 bool brathl::CFile::Open (const string & name, uint32_t mode = modeRead|typeBinary)

Opens a file. If file object is open, it is closed. If an error occurred, a **CFileException** (p. 243) is raised.

Parameters

<i>name</i>	[in] : full name of the file;
<i>mode</i>	[in] : access mode - default value : modeRead typeBinary (see openFlags (p. 238));

Returns

true on success, otherwise false

Referenced by GetStatus().

6.59.4.12 bool brathl::CFile::Open ()

Opens the current file object. If an error occurred, a **CFileException** (p. 243) is raised.

Returns

true on success, otherwise false

References BRATHL_IO_ERROR, and brathl::CTools::Format().

Referenced by brathl::CFileParams::Load().

6.59.4.13 int32_t brathl::CFile::ReadLine (char * lineRead, uint32_t size)

Function reads lines from the current file and places contents into buffer pointed by lineRead If an error occurred, a **CFileException** (p. 243) is raised.

Parameters

<i>lineRead</i>	[out] : line read
<i>size</i>	[in] : max number of bytes of the line

Returns

the number of bytes in the lineRead parameter. -1 if end of file reached

6.59.4.14 int32_t brathl::CFile::ReadLineData (char * lineRead, uint32_t size)

Same as **ReadLine** (p. 240), but reads only line of data and skip comments and places contents into buffer pointed by lineRead. Comments start with character '#' anywhere in the line. Empty line or space line are also skipped If an

error occurred, a **CFileException** (p. 243) is raised.

Parameters

<i>lineRead</i>	[out] : line data read
<i>size</i>	[in] : max number of bytes of the line

Returns

the number of bytes in the lineRead parameter. -1 if end of file reached

References brathl::CTools::Trim().

Referenced by brathl::CFileParams::Load(), and brathl::CMission::LoadAliasName().

6.59.4.15 `int32_t brathl::CFile::ReadToBuffer (char * destinationBuffer, uint32_t numBytesToRead = CFile::m_maxBufferToRead)`

Function reads 'NumBytesToRead' bytes from the current file position and places file contents into buffer pointed by destinationBuffer. If an error occurred, a **CFileException** (p. 243) is raised.

Parameters

<i>destinationBuffer</i>	[out] : destination buffer
<i>numBytesToRead</i>	[in] : number of bytes to reads

Returns

the number of bytes actually reads, zero if end of file reached

References BRATHL_IO_ERROR, and brathl::CTools::Format().

6.59.4.16 `bool brathl::CFile::Rename (const string & newName)`

Renames file object. If file with specified filename exists, its contents are erased. The current file is closed, renamed and opened as new name.

Parameters

<i>newName</i>	[in] : new file name
----------------	----------------------

Returns

true on success, otherwise false

6.59.4.17 `bool brathl::CFile::Rename (const string & oldName, const string & newName) [static]`

Renames a file. If file with specified filename exists, its contents are erased.

Parameters

<i>oldName</i>	[in] : file to rename
<i>newName</i>	[in] : new file name

Returns

true on success, otherwise false

6.59.4.18 `bool brathl::CFile::SeekToBegin ()`

Function moves moves file pointer to the beginning of file.

Returns

true on success, otherwise false

6.59.4.19 `bool brathl::CFile::SeekToEnd ()`

Function moves moves file pointer to the end of file.

Returns

true on success, otherwise false

6.59.4.20 `bool brathl::CFile::SetBufferingMode (bool mode = true)`

Change buffering mode. Function must be used before any read/write operation occurs!

Parameters

<i>mode</i>	[in] : true if buffered I/O (default), false if unbuffered I/O
-------------	--

Returns

true on success, otherwise false

6.59.4.21 `bool brathl::CFile::SetPosition (long positionOffset)`

Function moves file pointer by PositionOffset bytes relative to current position.

Parameters

<i>positionOffset</i>	[in] : offset to move
-----------------------	-----------------------

Returns

true on success, otherwise false

6.59.4.22 `bool brathl::CFile::WriteChar (const int character)`

Writes a single character to a file If an error occurred, a **CFileException** (p. 243) is raised.

Parameters

<i>character</i>	[in] : character to write
------------------	---------------------------

Returns

true on success, otherwise false

References BRATHL_IO_ERROR, and brathl::CTools::Format().

6.59.4.23 `uint32_t brathl::CFile::WriteFromBuffer (const char * sourceBuffer, uint32_t sourceBufferLength)`

Writes data from memory to a file If an error occurred, a **CFileException** (p. 243) is raised.

Parameters

<i>sourceBuffer</i>	[in] : data to write
<i>sourceBuffer- Length</i>	[in] : data length to write

Returns

the number of bytes actually written.

References BRATHL_IO_ERROR, and bratl::CTools::Format().

Referenced by Duplicate().

6.59.4.24 bool bratl::CFile::WriteString (const char * *str*)

Writes a string to a file. If an error occurred, a **CFileException** (p. 243) is raised.

Parameters

<i>str</i>	[in] : string to write
------------	------------------------

Returns

true on success, otherwise false

References BRATHL_IO_ERROR, and bratl::CTools::Format().

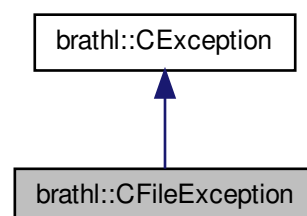
The documentation for this class was generated from the following files:

- File.h
- File.cpp

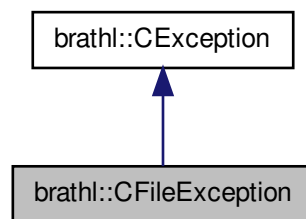
6.60 bratl::CFileException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for bratl::CFileException:



Collaboration diagram for brathl::CFileException:



Public Member Functions

- **CFileException** ()
*Empty **CFileException** (p. 243) ctor.*
- **CFileException** (const string &message, int32_t errcode=**BRATHL_ERROR**)
- **CFileException** (const string &message, const string &fileName, int32_t errcode)
- virtual const char * **TypeOf** () const
Identification of exception (human readable)
- virtual ~**CFileException** () throw ()
Destructor.

Additional Inherited Members

6.60.1 Detailed Description

File Exception management class.

Version

1.0

6.60.2 Constructor & Destructor Documentation

6.60.2.1 brathl::CFileException::CFileException (const string & message, int32_t errcode = **BRATHL_ERROR**)
[inline]

Creates a new **CFileException** (p. 243) object.

Parameters

<i>message</i>	[in] : error message
<i>errcode</i>	[in] : error code

6.60.2.2 brathl::CFileException::CFileException (const string & message, const string & fileName, int32_t errcode)

Creates a new **CFileException** (p. 243) object.

Parameters

<i>message</i>	[in] : error message
<i>fileName</i>	[in] : file name in error
<i>errcode</i>	[in] : error code

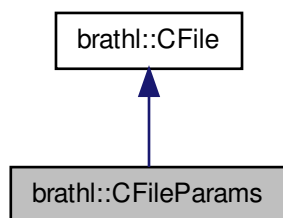
The documentation for this class was generated from the following files:

- **Exception.h**
- Exception.cpp

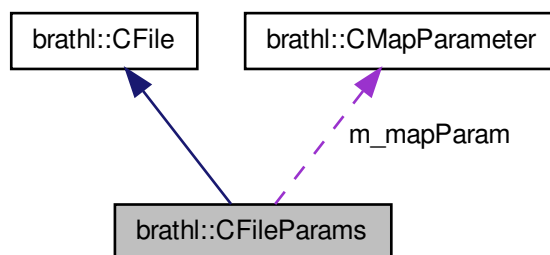
6.61 brathl::CFileParams Class Reference

```
#include <FileParams.h>
```

Inheritance diagram for brathl::CFileParams:



Collaboration diagram for brathl::CFileParams:



Public Member Functions

- **CFileParams ()**
Empty **CFileParams** (p. 245) ctor.
- **CFileParams** (const string &name, uint32_t mode=**modeRead**|**typeBinary**)
- uint32_t **CheckCount** (const string &Key, int32_t ValidMin=1, int32_t ValidMax=1)

- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- void **GetFieldNames** (const string &key, CStringArray &fieldNames)
- **CStringMap * GetFieldSpecificUnits** ()
- void **GetFileList** (const string &key, CStringArray &fileNames)
- string **GetFirstFile** (const string &key)
- bool **IsLoaded** ()
- void **Load** ()
- void **LoadAliases** ()
- void **LoadFieldSpecificUnits** ()
- void **SetVerboseLevel** ()
- void **SubstituteAliases** (const **CStringMap** &aliases)
- virtual **~CFileParams** ()
Destructor.
- void **Load** (const string &name, uint32_t mode=**modeRead|typeBinary**)

Public Attributes

- **CMapParameter m_mapParam**

Additional Inherited Members

6.61.1 Detailed Description

Parameters file management class.

This class provides ascii parameters file services

It makes it possible to acquire the whole of information which they contain

Parameters are described as 'keyword'='value'

keyword is character strings identifying a type of data. value is character strings associated with the key.

keyword and value have to be on the same line;

It don't make distinction between upper-case and lower-case letters.

While managing the file, if an error occurred, a **CFileException** (p.243) is raised. While managing parameter (keyword, value), if an error occurred, a **CParameterException** (p.278) is raised.

Version

1.0

6.61.2 Constructor & Destructor Documentation

6.61.2.1 brathl::CFileParams::CFileParams (const string & name, uint32_t mode = modeRead|typeBinary)

Creates new **CFileParams** (p.245) object and opens the parameters file. On error, a **CFileException** (p.243) or **CParameterException** (p.278) exception is raised.

Parameters

<i>name</i>	[in] : full name of the file;
<i>mode</i>	[in] : access mode - default value : modeRead typeBinary (see openFlags (p.238));

References Load().

6.61.3 Member Function Documentation

6.61.3.1 uint32_t brathl::CFileParams::CheckCount (const string & Key, int32_t ValidMin = 1, int32_t ValidMax = 1)

Throw an exception if the number of values is not valid.

Parameters

<i>ValidMin</i>	[in] : Minimal number of values
<i>ValidMax</i>	[in] : Maximal number of values. If <=0, it is considered as infinite. If < ValidMin, it is considered as equal to ValidMin.

Returns

actual number of occurrences of the parameter

References BRATHL_COUNT_ERROR, brathl::CParameter::Count(), and brathl::CTools::Format().

Referenced by SetVerboseLevel().

6.61.3.2 void brathl::CFileParams::Load ()

Reads file parameters and load parameters On error, a **CFileException** (p. 243) or **CParameterException** (p. 278) exception is raised.

References brathl::CFile::Close(), brathl::CFile::IsOpen(), m_mapParam, brathl::CFile::Open(), brathl::CFile::ReadLineData(), and brathl::CMapParameter::RemoveAll().

Referenced by CFileParams(), and Load().

6.61.3.3 void brathl::CFileParams::Load (const string & name, uint32_t mode = modeRead|typeBinary)

Reads file parameters and load parameters On error, a **CFileException** (p. 243) or **CParameterException** (p. 278) exception is raised.

Parameters

<i>name</i>	[in] : full name of the file;
<i>mode</i>	[in] : access mode - default value : modeRead typeBinary (see openFlags (p. 238));

References Load(), and brathl::CFile::Open().

6.61.3.4 void brathl::CFileParams::SetVerboseLevel ()

Set the verbosity level from the standard keyword VERBOSE

References CheckCount(), and m_mapParam.

6.61.4 Member Data Documentation

6.61.4.1 CMapParameter brathl::CFileParams::m_mapParam

A map containing all the parameters

Referenced by Dump(), Load(), and SetVerboseLevel().

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

- FileParams.h
- FileParams.cpp

6.62 brathl::CFloatArray Class Reference

```
#include <List.h>
```

Public Member Functions

- **CFloatArray** ()
Empty CFloatArray (p. 248) ctor.
- **CFloatArray** (const **CFloatArray** &vect)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CFloatArray::iterator it)
- void **GetRange** (float &min, float &max)
- virtual void **Insert** (float *data, int32_t size)
- virtual void **Insert** (const **CFloatArray** &vect, bool bEnd=true)
- virtual void **Insert** (const **CFloatArray** &vect, int32_t first, int32_t last, bool bEnd=true)
- virtual void **Insert** (const float value)
- virtual void **Insert** (const int32_t value)
- virtual void **Insert** (const uint32_t value)
- virtual CFloatArray::iterator **InsertAt** (CFloatArray::iterator where, const float value)
- virtual CFloatArray::iterator **InsertAt** (int32_t index, const float value)
- virtual bool **Intersect** (const **CFloatArray** &array, **CFloatArray** &intersect) const
- virtual const **CFloatArray** & **operator=** (const **CFloatArray** &vect)
- virtual void **RemoveAll** ()
- virtual CFloatArray::iterator **ReplaceAt** (CFloatArray::iterator where, const float value)
- virtual CFloatArray::iterator **ReplaceAt** (int32_t index, const float value)
- float * **ToArray** ()
- virtual string **ToString** (const string &delim=",", bool useBracket=true) const
- virtual ~**CFloatArray** ()
Destructor.

6.62.1 Detailed Description

An array (vector) of float management class.

Version

1.0

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

- List.h
- List.cpp

6.63 brathl::CProduct::CInfo Class Reference

```
#include <Product.h>
```

Inherits brathl::CBratObject.

Public Attributes

- string **m_fieldName**
- int32_t **m_index**
- int32_t **m_isUnion**
- coda_Type * **m_type**
- coda_type_class **m_type_class**

Additional Inherited Members

6.63.1 Detailed Description

A class to traverse Brat files

Version

1.0

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

- Product.h
- Product.cpp

6.64 brathl::CInt16Array Class Reference

```
#include <List.h>
```

Public Member Functions

- **CInt16Array** ()
Empty CInt16Array (p. 249) ctor.
- **CInt16Array** (const **CInt16Array** &vect)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CInt16Array::iterator it)
- virtual void **Insert** (const **CInt16Array** &vect, bool bEnd=true)
- virtual void **Insert** (const CStringArray &vect)
- virtual void **Insert** (int16_t *vect, size_t length)
- virtual void **Insert** (const int16_t value)
- virtual CInt16Array::iterator **InsertAt** (CInt16Array::iterator where, const int16_t value)
- virtual CInt16Array::iterator **InsertAt** (int32_t index, const int16_t value)
- virtual bool **Intersect** (const **CInt16Array** &array, **CInt16Array** &intersect) const
- virtual const **CInt16Array** & **operator=** (const **CInt16Array** &vect)
- virtual void **RemoveAll** ()
- virtual CInt16Array::iterator **ReplaceAt** (CInt16Array::iterator where, const int16_t value)
- virtual CInt16Array::iterator **ReplaceAt** (int32_t index, const int16_t value)
- virtual int16_t * **ToArray** ()
- virtual string **ToString** (const string &delim=",", bool useBracket=true) const
- virtual ~**CInt16Array** ()
Destructor.

6.64.1 Detailed Description

An array (vector) of ints management class.

Version

1.0

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

- List.h
- List.cpp

6.65 brathl::CInt64Array Class Reference

```
#include <List.h>
```

Public Member Functions

- **CInt64Array** ()
Empty CInt64Array (p. 250) ctor.
- **CInt64Array** (const **CInt64Array** &v)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CInt64Array::iterator it)
- virtual void **IncrementValue** (uint64_t incr=1)
- virtual void **Insert** (const **CInt64Array** &vect, bool bEnd=true)
- virtual void **Insert** (const CStringArray &vect)
- virtual void **Insert** (int64_t *vect, size_t length)
- virtual void **Insert** (const int64_t value)
- virtual CInt64Array::iterator **InsertAt** (CInt64Array::iterator where, const int64_t value)
- virtual CInt64Array::iterator **InsertAt** (size_t index, const int64_t value)
- virtual bool **Intersect** (const **CInt64Array** &array, **CInt64Array** &intersect) const
- virtual const **CInt64Array** & **operator=** (const **CInt64Array** &vect)
- virtual bool **operator==** (const **CInt64Array** &vect)
- virtual void **RemoveAll** ()
- virtual CInt64Array::iterator **ReplaceAt** (CInt64Array::iterator where, const int64_t value)
- virtual CInt64Array::iterator **ReplaceAt** (size_t index, const int64_t value)
- virtual int64_t * **ToArray** ()
- virtual string **ToString** (const string &delim=",", bool useBracket=true) const
- virtual ~**CInt64Array** ()
Destructor.

6.65.1 Detailed Description

An array (vector) of ints management class.

Version

1.0

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

- List.h
- List.cpp

6.66 brathl::CInt8Array Class Reference

```
#include <List.h>
```

Public Member Functions

- **CInt8Array** ()
*Empty **CInt8Array** (p. 251) ctor.*
- **CInt8Array** (const **CInt8Array** &vect)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CInt8Array::iterator it)
- virtual void **Insert** (const **CInt8Array** &vect, bool bEnd=true)
- virtual void **Insert** (const CStringArray &vect)
- virtual void **Insert** (int8_t *vect, size_t length)
- virtual void **Insert** (const int8_t value)
- virtual CInt8Array::iterator **InsertAt** (CInt8Array::iterator where, const int8_t value)
- virtual CInt8Array::iterator **InsertAt** (int32_t index, const int8_t value)
- virtual bool **Intersect** (const **CInt8Array** &array, **CInt8Array** &intersect) const
- virtual const **CInt8Array** & **operator=** (const **CInt8Array** &vect)
- virtual void **RemoveAll** ()
- virtual CInt8Array::iterator **ReplaceAt** (CInt8Array::iterator where, const int8_t value)
- virtual CInt8Array::iterator **ReplaceAt** (int32_t index, const int8_t value)
- virtual int8_t * **ToArray** ()
- virtual string **Tostring** (const string &delim=",", bool useBracket=true) const
- virtual ~**CInt8Array** ()
Destructor.

6.66.1 Detailed Description

An array (vector) of ints management class.

Version

1.0

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

- List.h
- List.cpp

6.67 brathl::CIntArray Class Reference

```
#include <List.h>
```

Public Member Functions

- **CIntArray** ()
*Empty **CIntArray** (p. 251) ctor.*
- **CIntArray** (const **CIntArray** &vect)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CIntArray::iterator it)

- virtual void **IncrementValue** (uint32_t incr=1)
- virtual void **Insert** (const **CIntArray** &vect, bool bEnd=true)
- virtual void **Insert** (const CStringArray &vect)
- virtual void **Insert** (int32_t *vect, size_t length)
- virtual void **Insert** (const int32_t value)
- virtual CIntArray::iterator **InsertAt** (CIntArray::iterator where, const int32_t value)
- virtual CIntArray::iterator **InsertAt** (int32_t index, const int32_t value)
- virtual bool **Intersect** (const **CIntArray** &array, **CIntArray** &intersect) const
- virtual const **CIntArray** & **operator=** (const **CIntArray** &vect)
- virtual bool **operator==** (const **CIntArray** &vect)
- virtual void **RemoveAll** ()
- virtual CIntArray::iterator **ReplaceAt** (CIntArray::iterator where, const int32_t value)
- virtual CIntArray::iterator **ReplaceAt** (int32_t index, const int32_t value)
- virtual int32_t * **ToArray** ()
- virtual string **ToString** (const string &delim=",", bool useBracket=true) const
- virtual ~**CIntArray** ()

Destructor.

6.67.1 Detailed Description

An array (vector) of ints management class.

Version

1.0

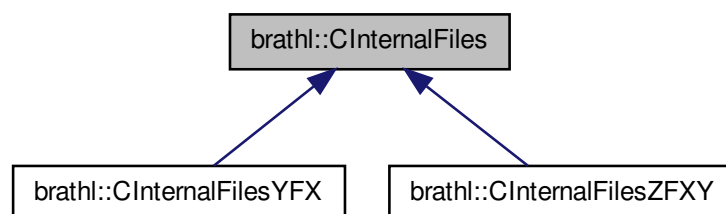
The documentation for this class was generated from the following files:

- List.h
- List.cpp

6.68 brathl::CInternalFiles Class Reference

```
#include <InternalFiles.h>
```

Inheritance diagram for brathl::CInternalFiles:



Public Member Functions

- CNetCDFDimension * **AddNetCDFDim** (CNetCDFDimension &dim, bool forceReplace=false)
- CNetCDFVarDef * **AddNetCDFVarDef** (CNetCDFVarDef &var, bool forceReplace=false)
- **CInternalFiles** (string Name="", **brathl_FileMode** Mode=ReadOnly)
- virtual void **Close** ()
- int **GetAttribute** (const string &varName, const string &attName, double &attValue, bool mustExist=true, double defaultValue=**CTools::m_defaultValueDOUBLE**)
- int **GetAttribute** (const string &varName, const string &attName, string &attValue, bool mustExist=true, string defaultValue="")
- virtual void **GetAxisVars** (vector< string > &VarNames)
- string **GetComment** (const string &varName)
- virtual bool **GetCommonVarDims** (const string &varName1, const string &varName2, CStringArray &intersect)
- virtual bool **GetComplementVarDims** (const string &varName1, const string &varName2, CStringArray &complement)
- virtual bool **GetComplementVars** (const CStringArray &varNames, CStringArray &complement, bool excludeDim=true)
- virtual void **GetDataVars** (vector< string > &VarNames)
- int **GetDimId** (const string &name)
- CNetCDFFiles * **GetFile** ()
- uint32_t **GetMaxFieldNumberOfDims** (const CStringArray *fieldNames=NULL)
- virtual string **GetName** () const
- CNetCDFDimension * **GetNetCDFDim** (const string &name)
- **CObMap** * **GetNetCDFDims** ()
- void **GetNetCDFDims** (const string &varName, **CObArray** *dims)
- CNetCDFVarDef * **GetNetCDFVarDef** (const string &name)
- **CObMap** * **GetNetCDFVarDefs** ()
- virtual string **GetTitle** (const string &Name)
- virtual string **GetType** ()
- virtual CUnit **GetUnit** (const string &Name)
- int32_t **GetVarDimIndex** (const string &varName, const string &dimName)
- virtual void **GetVarDims** (const string &Name, **ExpressionValueDimensions** &Dimensions)
- virtual void **GetVarDims** (const string &Name, vector< string > &Dimensions)
- virtual void **GetVariables** (vector< string > &VarNames)
- virtual NetCDFVarKind **GetVarKind** (const string &Name)
- virtual bool **HasVar** (NetCDFVarKind VarKind)
- bool **IsAxisVar** (const string &Name)
- virtual bool **IsGeographic** ()
- virtual bool **IsOpened** ()
- virtual void **Open** (**brathl_FileMode** mode)
- virtual void **Open** ()
- virtual void **ReadVar** (const string &Name, **CExpressionValue** &Value, const string &WantedUnit)
- void **ReplaceNetCDFDim** (CNetCDFDimension &dim)
- virtual void **SetMode** (**brathl_FileMode** mode)
- virtual void **SetName** (const string &name)
- virtual bool **VarExists** (const string &Name)
- virtual void **WriteData** (CNetCDFVarDef *varDef, **CExpressionValue** *data)
- virtual void **WriteData** (CNetCDFVarDef *varDef, CMatrix *matrix)
- virtual void **WriteDimensions** ()
- virtual void **WriteFileTitle** (const string &Title)
- virtual void **WriteVar** (const string &Name, const **CExpressionValue** &Value)
- virtual void **WriteVariables** ()

Static Public Member Functions

- static **CInternalFiles** * **Create** (const string &fileName, bool open=true, bool withExcept=true)
- static bool **IsVarNameValid** (const string &Name)
- static bool **IsYFXFile** (const string &fileName, **CInternalFiles** **pf=NULL)
- static bool **IsYFXFile** (**CInternalFiles** *f, CStringArray *fieldNamesIn=NULL)
- static bool **IsZFlatLonFile** (const string &fileName, **CInternalFiles** **pf=NULL)
- static bool **IsZFlatLonFile** (**CInternalFiles** *f)
- static bool **IsZFXFile** (const string &fileName, CStringArray *fieldNames=NULL, **CInternalFiles** **pf=NULL)
- static bool **IsZFXFile** (**CInternalFiles** *f, CStringArray *fieldNames=NULL)
- static string **TypeOf** ()

Protected Member Functions

- void **SetFixedGlobalAttributes** (void)

Protected Attributes

- CNetCDFFiles **m_file**

6.68.1 Detailed Description

Internal files access.

Version

1.0

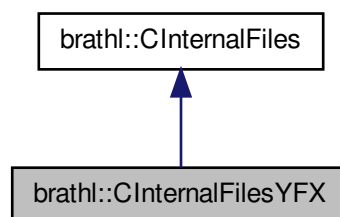
The documentation for this class was generated from the following files:

- InternalFiles.h
- InternalFiles.cpp

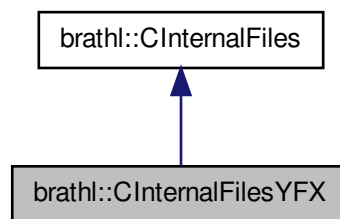
6.69 bratl::CInternalFilesYFX Class Reference

```
#include <InternalFilesYFX.h>
```

Inheritance diagram for bratl::CInternalFilesYFX:



Collaboration diagram for brathl::CInternalFilesYFX:



Public Member Functions

- **CInternalFilesYFX** (string Name="", **brathl_FileMode** Mode=ReadOnly)
- virtual void **CreateData** (const string &Name, const string &Units, const string &LongName, const string &Comment="", double ValidMin=**CTools::m_defaultValueDOUBLE**, double ValidMax=**CTools::m_defaultValueDOUBLE**, nc_type Type=NC_DOUBLE)
- virtual void **CreateDim** (NetCDFVarKind Kind, const string &XName, const **CExpressionValue** &Values, const string &Units, const string &LongName, const string &Comment="", double ValidMin=**CTools::m_defaultValueDOUBLE**, double ValidMax=**CTools::m_defaultValueDOUBLE**)
- virtual string **GetType** ()

Static Public Member Functions

- static string **TypeOf** ()

Additional Inherited Members

6.69.1 Detailed Description

Internal files access for internal files used to store $Y=F(X)$ kind of data.

Version

1.0

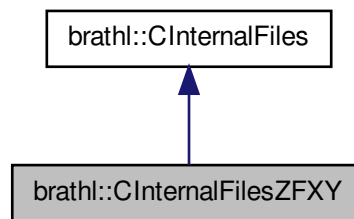
The documentation for this class was generated from the following files:

- InternalFilesYFX.h
- InternalFilesYFX.cpp

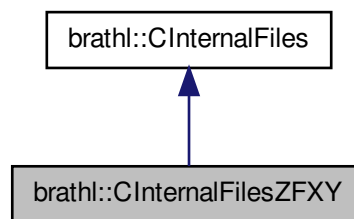
6.70 brathl::CInternalFilesZFXY Class Reference

```
#include <InternalFilesZFXY.h>
```

Inheritance diagram for bratl::CInternalFilesZFX Y:



Collaboration diagram for bratl::CInternalFilesZFX Y:



Public Member Functions

- **CInternalFilesZFX Y** (string Name="", **bratl_FileMode** Mode=ReadOnly)
- virtual void **CreateData** (const string &Name, const string &Units, const string &LongName, const string &Dim1Name, const string &Dim2Name, const string &Comment="", double ValidMin=**CTools::m_defaultValueDOUBLE**, double ValidMax=**CTools::m_defaultValueDOUBLE**, nc_type Type=NC_DOUBLE)
- virtual void **CreateDim** (NetCDFVarKind Kind, const string &XName, const **CExpressionValue** &Values, const string &Units, const string &LongName, const string &Comment="", double ValidMin=**CTools::m_defaultValueDOUBLE**, double ValidMax=**CTools::m_defaultValueDOUBLE**)
- virtual string **GetType** ()
- virtual bool **IsGeographic** ()

Static Public Member Functions

- static string **TypeOf** ()

Additional Inherited Members

6.70.1 Detailed Description

Internal files access for internal files used to store $Y=F(X)$ kind of data.

Version

1.0

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

- InternalFilesZFX.Y.h
- InternalFilesZFX.Y.cpp

6.71 brathl::CIntList Class Reference

```
#include <List.h>
```

Public Member Functions

- **CIntList** ()
*Empty **CIntList** (p. 257) ctor.*
- **CIntList** (const **CIntList** &list)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual void **Insert** (const **CIntList** &list, bool bEnd=true)
- virtual void **Insert** (const int value, bool bEnd=true)
- const **CIntList** & **operator=** (const **CIntList** &lst)
- virtual void **RemoveAll** ()
- virtual ~**CIntList** ()
Destructor.

6.71.1 Detailed Description

A list of strings management class.

Version

1.0

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

- List.h
- List.cpp

6.72 brathl::CIntMap Class Reference

```
#include <List.h>
```

Public Member Functions

- **CIntMap** ()
***CIntMap** (p. 257) ctor.*
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CIntMap::iterator it)
- virtual bool **Erase** (const string &key)
- virtual int32_t **Exists** (const string &key) const

- virtual int32_t **Insert** (const string &key, int32_t value, bool withExcept=true)
- virtual void **Insert** (const **CIntMap** &m, bool bRemoveAll=true, bool withExcept=true)
- virtual void **Insert** (const CStringArray &keys, const **CIntArray** &values, bool bRemoveAll=true, bool withExcept=true)
- virtual int32_t **operator[]** (const string &key)
- virtual void **RemoveAll** ()
- virtual ~**CIntMap** ()

CIntMap (p. 257) *dtor.*

6.72.1 Detailed Description

a set of integer value management classes.

Version

1.0

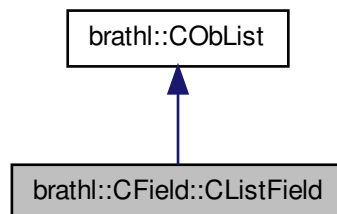
The documentation for this class was generated from the following files:

- List.h
- List.cpp

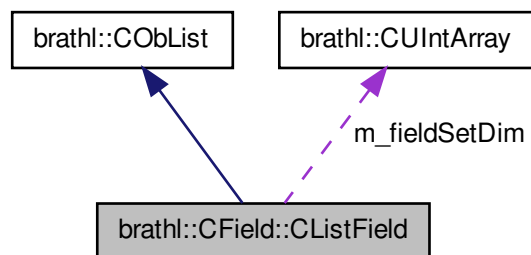
6.73 brathl::CField::CListField Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CField::CListField:



Collaboration diagram for bratl::CField::CListField:



Public Member Functions

- **CField * Back** (bool withExcept=true)
- **CListField** (bool bDelete)
- **CField * Front** (bool withExcept=true)
- virtual void **InsertField** (**CField** *field, bool hasDataset=true, bool bEnd=true)
- void **RemoveAll** ()

Public Attributes

- **CUIntArray m_fieldSetDim**
- **int32_t m_nbFieldSetDims**

Additional Inherited Members

6.73.1 Detailed Description

A list of **CField** (p. 208) object management class

Version

1.0

6.73.2 Member Function Documentation

6.73.2.1 void bratl::CField::CListField::RemoveAll () [virtual]

Remove all elements and clear the list

Reimplemented from **bratl::CObList** (p. 62).

References **bratl::CObList::RemoveAll()**.

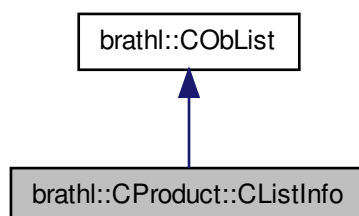
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

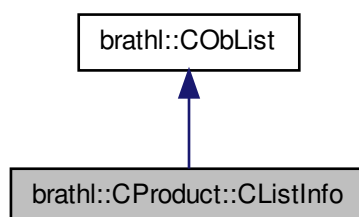
6.74 bratl::CProduct::CListInfo Class Reference

```
#include <Product.h>
```

Inheritance diagram for bratl::CProduct::CListInfo:



Collaboration diagram for bratl::CProduct::CListInfo:



Public Member Functions

- **CInfo * AddNew** ()
- **CInfo * Back** (bool withExcept=true)
- **CInfo * Front** (bool withExcept=true)
- **CInfo * PrevBack** (bool withExcept=true)

Additional Inherited Members

6.74.1 Detailed Description

A list of **CInfo** (p. 248) object management class

Version

1.0

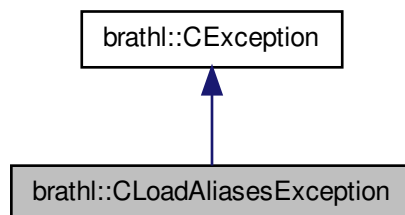
The documentation for this class was generated from the following files:

- Product.h
- Product.cpp

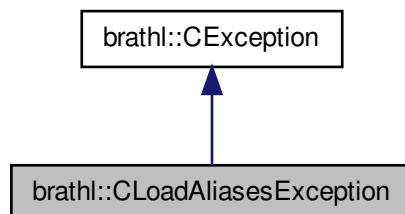
6.75 bratl::CLoadAliasesException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for bratl::CLoadAliasesException:



Collaboration diagram for bratl::CLoadAliasesException:



Public Member Functions

- **CLoadAliasesException** (const string &message, int32_t errcode)
- virtual const char * **TypeOf** () const
Identification of exception (human readable)
- virtual ~**CLoadAliasesException** () throw ()
Destructor.

Additional Inherited Members

6.75.1 Detailed Description

Aliases loading Exception management class.

Version

1.0

6.75.2 Constructor & Destructor Documentation

6.75.2.1 brathl::CLoadAliasesException::CLoadAliasesException (const string & message, int32_t errcode) [inline]

Creates a new **CParameterException** (p. 278) object.

Parameters

<i>message</i>	[in] : error message
<i>errcode</i>	[in] : error code

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

- **Exception.h**

6.76 brathl::CMapParameter Class Reference

```
#include <MapParameter.h>
```

Public Member Functions

- **CMapParameter** ()
CMapParameter (p. 262) ctor.
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- bool **Erase** (CMapParameter::iterator iteratorParameter)
- bool **Erase** (const string &key)
- **CParameter** * **Exists** (const string &key)
- **CParameter** * **Insert** (const string &key, const string &value)
- **CParameter** * **operator[]** (const string key)
- void **RemoveAll** ()
- virtual ~**CMapParameter** ()
CMapParameter (p. 262) dtor.

6.76.1 Detailed Description

Parameter management class.

This class provides a map of **CParameter** (p. 275) objects

Version

1.0

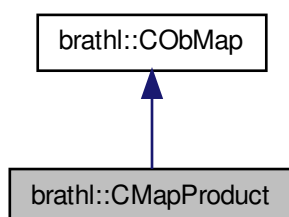
The documentation for this class was generated from the following files:

- **MapParameter.h**
- **MapParameter.cpp**

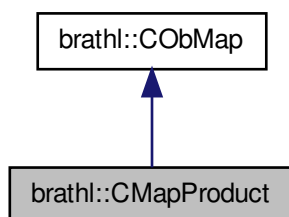
6.77 brathl::CMapProduct Class Reference

```
#include <Product.h>
```

Inheritance diagram for brathl::CMapProduct:



Collaboration diagram for brathl::CMapProduct:



Public Member Functions

- void **AddCriteriaToProducts** ()
- **CMapProduct** ()
CIntMap (p. 257) *ctor.*
- virtual void **Dump** (ostream &fOut=cerr)
- void **GetProductKeysWithCriteria** (CStringArray &keys)
- void **RemoveCriteriaFromProducts** ()
- virtual **~CMapProduct** ()
CIntMap (p. 257) *dtor.*

Static Public Member Functions

- static **CMapProduct &GetInstance** ()

Protected Member Functions

- void **Init** ()

Additional Inherited Members

6.77.1 Detailed Description

Mapping products management class.

Version

1.0

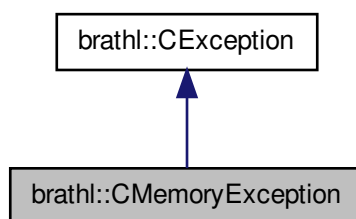
The documentation for this class was generated from the following files:

- Product.h
- Product.cpp

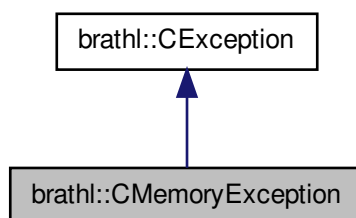
6.78 bratl::CMemoryException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for bratl::CMemoryException:



Collaboration diagram for bratl::CMemoryException:



Public Member Functions

- **CMemoryException** ()

Empty **CMemoryException** (p. 264) ctor.

- **CMemoryException** (const string &message, int32_t errcode=**BRATHL_MEMORY_ERROR**)
 - virtual const char * **TypeOf** () const
- Identification of exception (human readable)
- virtual ~**CMemoryException** () throw ()

Destructor.

Additional Inherited Members

6.78.1 Detailed Description

memory Exception management class.

Version

1.0

6.78.2 Constructor & Destructor Documentation

6.78.2.1 brathl::CMemoryException::CMemoryException (const string & message, int32_t errcode = **BRATHL_MEMORY_ERROR**) [inline]

Creates a new **CMemoryException** (p. 264) object.

Parameters

<i>message</i>	[in] : error message
<i>errcode</i>	[in] : error code

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

- **Exception.h**

6.79 brathl::CMission Class Reference

```
#include <Mission.h>
```

Public Member Functions

- **CMission** (**brathl_mission** mission, bool printWarnings=true)
 - **CMission** (**brathl_mission** mission, const double repeat, const **CDate** &dateRef, const uint32_t cycleRef, const uint32_t passRef, const uint32_t nbPass, bool printWarnings=true)
 - int32_t **Convert** (**CDate** &date, uint32_t &cycle, uint32_t &pass)
 - int32_t **Convert** (uint32_t cycle, uint32_t pass, **CDate** &date)
 - int32_t **CtrlMission** ()
 - virtual void **Dump** (ostream &fOut=cerr)
- Dump fonction.
- uint32_t **GetCycleRef** ()
 - const **CDate** & **GetDateRef** ()
 - **brathl_mission** **GetMission** ()
 - const char *const **GetName** ()
 - uint32_t **GetNbPass** ()
 - uint32_t **GetPassRef** ()
 - double **GetRepeat** ()

- int32_t **LoadAliasName** (CStringList &aliases)
- const **CMission** & **operator=** (const **CMission** &m)

Static Public Member Functions

- static double **GetGlobalConstant** (brathl_global_constants constantValue)

Static Public Attributes

- static const int **m_maxLenName** = 30
- static const char * **m_nameE2** = "ERS2"
- static const char * **m_nameE_C** = "ERS1-A"
- static const char * **m_nameE_G** = "ERS1-B"
- static const char * **m_nameEN** = "ENVISAT"
- static const char * **m_nameG2** = "GFO"
- static const char * **m_nameJ1** = "Jason-1"
- static const char * **m_nameJ2** = "Jason-2"
- static const char * **m_nameTP** = "TopeX/Poseidon"
- static const char * **m_nameUnknown** = "Unknown mission"
- static const char * **m_refAliasName** = "brathl_aliasmission.txt"
- static const char * **m_refFileName** = "brathl_refmission.txt"

6.79.1 Detailed Description

Satellite cycle/date conversion class.

A class to convert a date in a satellite cycle and pass number, or vice versa

Version

1.0

6.79.2 Constructor & Destructor Documentation

6.79.2.1 brathl::CMission::CMission (brathl_mission mission, bool printWarnings = true)

Constructs a **CMission** (p. 265) object

Parameters

<i>mission</i>	[in] : mission type (see brathl_mission (p. 372))
<i>printWarnings</i>	[in] : set to true for printing warnings on standard output, false otherwise. Default value is true.

References BRATHL_ERROR_INVALID_MISSION, and BRATHL_SUCCESS.

6.79.2.2 brathl::CMission::CMission (brathl_mission mission, const double repeat, const CDate & dateRef, const uint32_t cycleRef, const uint32_t passRef, const uint32_t nbPass, bool printWarnings = true)

Constructs a **CMission** (p. 265) object

Parameters

<i>mission</i>	[in] : mission type (see brathl_mission (p. 372))
<i>repeat</i>	[in] : duration that takes the satellite to return at the same point
<i>dateRef</i>	[in] : date reference in decimal julian day
<i>cycleRef</i>	[in] : cycle reference
<i>passRef</i>	[in] : pass reference

<i>nbPass</i>	[in] : numbers of half passes in a cycle
<i>printWarnings</i>	[in] : set to true for printing warnings on standard output, false otherwise. Default value is true.

6.79.3 Member Function Documentation

6.79.3.1 `int32_t brathl::CMission::Convert (CDate & date, uint32_t & cycle, uint32_t & pass)`

Converts a **CDate** (p. 173) object into acycle/pass

Parameters

<i>date</i>	[in] : date to convert
<i>cycle</i>	[out] : number of cycle
<i>pass</i>	[out] : number of pass in the cycle

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Cycle/date conversion error codes** (p. 19))

References BRATHL_ERROR_INVALID_NB_PASS, BRATHL_ERROR_INVALID_REPETITION, BRATHL_SUCCESS, and brathl::CDate::Convert2DecimalJulian().

Referenced by brathl_Cycle2YMDHMSM(), and brathl_YMDHMSM2Cycle().

6.79.3.2 `int32_t brathl::CMission::Convert (uint32_t cycle, uint32_t pass, CDate & date)`

Converts a cyle/pass into a **CDate** (p. 173) object

Parameters

<i>cycle</i>	[in] : number of cycle to convert
<i>pass</i>	[in] : number of pass in the cycle to cinvert
<i>date</i>	[out] : date corresponding to the cycle/pass

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Cycle/date conversion error codes** (p. 19))

References BRATHL_SUCCESS, and brathl::CDate::SetDateJulian().

6.79.3.3 `int32_t brathl::CMission::CtrlMission ()`

Tests if the mission is valid

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Cycle/date conversion error codes** (p. 19))

References BRATHL_ERROR_INVALID_MISSION, BRATHL_SUCCESS, ENVISAT, ERS1_A, ERS1_B, ERS2, G-FO, JASON1, JASON2, and TOPEX.

Referenced by brathl_Cycle2YMDHMSM(), and brathl_YMDHMSM2Cycle().

6.79.3.4 `uint32_t brathl::CMission::GetCycleRef () [inline]`

Gets the cycle reference attributes (see #m_cycleRef)

6.79.3.5 `const CDate& brathl::CMission::GetDateRef () [inline]`

Gets the date reference attributes (see #m_dateRef)

6.79.3.6 brathl_mission brathl::CMission::GetMission () [inline]

Gets the mission (see **brathl_mission** (p. 372))

6.79.3.7 const char *const brathl::CMission::GetName ()

Gets the name of the mission

References ENVISAT, ERS1_A, ERS1_B, ERS2, GFO, JASON1, JASON2, m_nameE2, m_nameE_C, m_nameE_G, m_nameEN, m_nameG2, m_nameJ1, m_nameJ2, m_nameTP, m_nameUnknown, and TOPEX.

6.79.3.8 uint32_t brathl::CMission::GetNbPass () [inline]

Gets the number of passes attributes (see #m_nbPass)

6.79.3.9 uint32_t brathl::CMission::GetPassRef () [inline]

Gets the pass reference attributes (see #m_passRef)

6.79.3.10 double brathl::CMission::GetRepeat () [inline]

Gets the repeat attributes (see #m_repeat)

6.79.3.11 int32_t brathl::CMission::LoadAliasName (CStringList & aliases)

Gets aliases names for the mission

Parameters

<i>aliases</i>	[out] : aliases for the mission
----------------	---------------------------------

Returns

BRATHL_SUCCESS (p. 17) or error code (see **Cycle/date conversion error codes** (p. 19))

References BRATHL_SUCCESS, BRATHL_WARNING_INVALID_REF_FILE_FIELD, BRATHL_WARNING_OPEN_FILE_ALIAS_MISSION, brathl::CFile::Close(), brathl::CTools::FindDataFile(), brathl::CTools::GetDataDir(), brathl::CFile::IsOpen(), m_refAliasName, brathl::CFile::modeRead, brathl::CFile::ReadLineData(), and brathl::CTools::StringTrim().

6.79.3.12 const CMission & brathl::CMission::operator= (const CMission & m)

Assigns a new value to the **CMission** (p. 265) object, with a **CMission** (p. 265) object

6.79.4 Member Data Documentation**6.79.4.1 const int brathl::CMission::m_maxLenName = 30 [static]**

Max length of the name of the mission

6.79.4.2 const char * brathl::CMission::m_nameE2 = "ERS2" [static]

Name of the ERS2 mission

Referenced by GetName().

6.79.4.3 const char * brathl::CMission::m_nameE_C = "ERS1-A" [static]

Name of the ERS1-A mission

Referenced by GetName().

6.79.4.4 `const char * bratl::CMission::m_nameE_G = "ERS1-B" [static]`

Name of the ERS1-B mission

Referenced by GetName().

6.79.4.5 `const char * bratl::CMission::m_nameEN = "ENVISAT" [static]`

Name of the ENVISAT mission

Referenced by GetName().

6.79.4.6 `const char * bratl::CMission::m_nameG2 = "GFO" [static]`

Name of the GFO mission

Referenced by GetName().

6.79.4.7 `const char * bratl::CMission::m_nameJ1 = "Jason-1" [static]`

Name of the Jason-1 mission

Referenced by GetName().

6.79.4.8 `const char * bratl::CMission::m_nameJ2 = "Jason-2" [static]`

Name of the Jason-2 mission

Referenced by GetName().

6.79.4.9 `const char * bratl::CMission::m_nameTP = "Topex/Poseidon" [static]`

Name of the Topex/Poseidon mission

Referenced by GetName().

6.79.4.10 `const char * bratl::CMission::m_nameUnknown = "Unknown mission" [static]`

Name of an unknown mission

Referenced by GetName().

6.79.4.11 `const char * bratl::CMission::m_refAliasName = "bratl_aliasmission.txt" [static]`

Name of the mission aliases file

An ascii file with records : field 1 : Name of the mission field 2 : Alias of the mission

Each field has to be separated by at least a non-numeric character

The file can contained several record for a same mission.

Referenced by LoadAliasName().

6.79.4.12 `const char * bratl::CMission::m_refFileName = "bratl_refmission.txt" [static]`

Name of the mission reference file

An ascii file with records : field 1 : Name of the mission field 2 : cycle reference field 3 : pass reference field 4 : date reference in decimal julian day

Each field has to be separated by at least a non-numeric character

The file can contained several record for a same mission. Only the field with the greatest date is taken into account

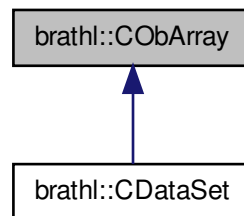
The documentation for this class was generated from the following files:

- Mission.h
- Mission.cpp

6.80 brathl::CObArray Class Reference

```
#include <List.h>
```

Inheritance diagram for brathl::CObArray:



Public Member Functions

- **CObArray** (bool bDelete=true)
Empty CObArray (p. 270) ctor.
- **CObArray** (const **CObArray** &vect)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- bool **Erase** (CBratObject *ob)
- virtual bool **Erase** (CObArray::iterator it)
- virtual bool **Erase** (int32_t index)
- bool **GetDelete** ()
- virtual void **Insert** (const **CObArray** &vect)
- virtual void **Insert** (CBratObject *ob)
- virtual CObArray::iterator **InsertAt** (CObArray::iterator where, CBratObject *ob)
- virtual const **CObArray** & **operator=** (const **CObArray** &lst)
- virtual bool **PopBack** ()
- virtual void **RemoveAll** ()
- virtual CObArray::iterator **ReplaceAt** (CObArray::iterator where, CBratObject *ob)
- void **SetDelete** (bool value)
- virtual ~**CObArray** ()
Destructor.

Protected Attributes

- bool **m_bDelete**

6.80.1 Detailed Description

An array (vector) of CBratObject management class.

Version

1.0

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

- List.h
- List.cpp

6.81 brathl::CObDoubleMap Class Reference

```
#include <List.h>
```

Public Member Functions

- **CObDoubleMap** (bool bDelete=true)
CObMap (p. 274) ctor.
- virtual void **Dump** (ostream &fOut=cerr) const
Dump function.
- virtual bool **Erase** (CObDoubleMap::iterator it)
- virtual bool **Erase** (double key)
- virtual CBratObject * **Exists** (double key) const
- bool **GetDelete** ()
- virtual void **GetKeys** (CDoubleArray &keys, bool bRemoveAll=true)
- virtual CBratObject * **Insert** (double key, CBratObject *ob, bool withExcept=true)
- virtual void **Insert** (const **CObDoubleMap** &obMap, bool withExcept=true)
- virtual const **CObDoubleMap** & **operator=** (const **CObDoubleMap** &obMap)
- virtual CBratObject * **operator[]** (double key)
- virtual void **RemoveAll** ()
- bool **RenameKey** (double oldKey, double newKey)
- void **SetDelete** (bool value)
- virtual ~**CObDoubleMap** ()
CObMap (p. 274) dtor.

Protected Attributes

- bool **m_bDelete**

6.81.1 Detailed Description

a set of object management classes.

Version

1.0

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

- List.h
- List.cpp

6.82 brathl::CObIntMap Class Reference

```
#include <List.h>
```

Public Member Functions

- **COblntMap** (bool bDelete=true)
CObMap (p. 274) *ctor.*
- virtual void **Dump** (ostream &fOut=cerr) const
Dump function.
- virtual bool **Erase** (COblntMap::iterator it)
- virtual bool **Erase** (int32_t key)
- virtual CBratObject * **Exists** (int32_t key) const
- bool **GetDelete** ()
- virtual void **GetKeys** (CIntArray &keys, bool bRemoveAll=true)
- virtual CBratObject * **Insert** (int32_t key, CBratObject *ob, bool withExcept=true)
- virtual void **Insert** (const **COblntMap** &obMap, bool withExcept=true)
- virtual const **COblntMap** & **operator=** (const **COblntMap** &obMap)
- virtual CBratObject * **operator[]** (int32_t key)
- virtual void **RemoveAll** ()
- bool **RenameKey** (int32_t oldKey, int32_t newKey)
- void **SetDelete** (bool value)
- virtual ~**COblntMap** ()
CObMap (p. 274) *dtor.*

Protected Attributes

- bool **m_bDelete**

6.82.1 Detailed Description

a set of object management classes.

Version

1.0

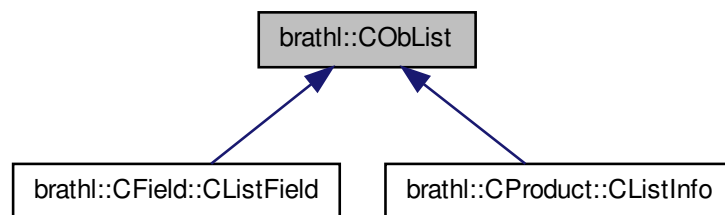
The documentation for this class was generated from the following files:

- List.h
- List.cpp

6.83 brathl::CObList Class Reference

```
#include <List.h>
```

Inheritance diagram for bratl::CObList:



Public Member Functions

- **CObList** (bool bDelete=true)
Empty CObList (p. 272) ctor.
- **CObList** (const **CObList** &lst)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump function.
- bool **Erase** (CBratObject *ob)
- virtual bool **Erase** (CObList::iterator it)
- bool **GetDelete** ()
- virtual void **Insert** (const **CObList** &list, bool bEnd=true)
- virtual void **Insert** (CBratObject *ob, bool bEnd=true)
- virtual const **CObList** & **operator=** (const **CObList** &lst)
- virtual bool **PopBack** ()
- virtual void **RemoveAll** ()
- void **SetDelete** (bool value)
- virtual ~**CObList** ()
Destructor.

Protected Attributes

- bool **m_bDelete**

6.83.1 Detailed Description

A list of CBratObject management class.

Version

1.0

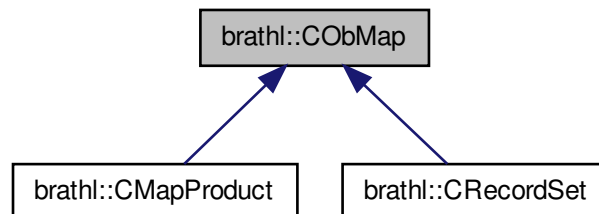
The documentation for this class was generated from the following files:

- List.h
- List.cpp

6.84 bratl::CObMap Class Reference

```
#include <List.h>
```

Inheritance diagram for bratl::CObMap:



Public Member Functions

- **CObMap** (bool bDelete=true)
CObMap (p. 274) ctor.
- **CObMap** (const **CObMap** &obMap)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump function.
- virtual bool **Erase** (CObMap::iterator it)
- virtual bool **Erase** (const string &key)
- virtual CBratObject * **Exists** (const string &key) const
- bool **GetDelete** ()
- virtual void **GetKeys** (CStringArray &keys, bool bRemoveAll=true, bool bUnique=false)
- virtual void **GetKeys** (CStringList &keys, bool bRemoveAll=true, bool bUnique=false)
- virtual CBratObject * **Insert** (const string &key, CBratObject *ob, bool withExcept=true)
- virtual void **Insert** (const **CObMap** &obMap, bool withExcept=true)
- virtual const **CObMap** & **operator=** (const **CObMap** &obMap)
- virtual CBratObject * **operator[]** (const string &key)
- virtual void **RemoveAll** ()
- bool **RenameKey** (const string &oldKey, const string &newKey)
- void **SetDelete** (bool value)
- virtual void **ToArray** (CObArray &obArray)
- virtual ~**CObMap** ()
CObMap (p. 274) dtor.

Protected Attributes

- bool **m_bDelete**

6.84.1 Detailed Description

a set of object management classes.

Version

1.0

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

- List.h
- List.cpp

6.85 brathl::CObStack Class Reference

```
#include <List.h>
```

Public Member Functions

- **CObStack** (bool bDelete=true)
Empty CObArray (p. 270) ctor.
- bool **GetDelete** ()
- virtual void **Pop** ()
- virtual void **Push** (CBratObject *ob)
- virtual void **RemoveAll** ()
- void **SetDelete** (bool value)
- virtual CBratObject * **Top** ()
- virtual ~**CObStack** ()
Destructor.

Protected Attributes

- bool **m_bDelete**
Dump fonction.

6.85.1 Detailed Description

An stack of CBratObject management class.

Version

1.0

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

- List.h
- List.cpp

6.86 brathl::CParameter Class Reference

```
#include <Parameter.h>
```

Public Member Functions

- uint32_t **Count** ()
- **CParameter** ()
 - Empty CParameter (p. 275) ctor.*
- **CParameter** (const char *keyword)
- virtual void **Dump** (ostream &fOut=cerr)
 - Dump function.*
- void **GetValue** (char *value, int32_t bufferSize, int32_t pos=0, const char *DefValue="")
- bool **RemoveAllValue** ()
- bool **RemoveValue** (uint32_t i)
- void **SetAliases** (const CStringMap &aliases)
- virtual ~**CParameter** ()
 - Destructor.*
- **CParameter** (const char *keyword, const char *value)
- **CParameter** (const string &keyword, const string &value)
- void **AddValue** (const char *value)
- void **AddValue** (const string &value)
- void **GetValue** (int32_t &value, int32_t pos=0, int32_t DefValue=CTools::m_defaultValueINT32)
- void **GetValue** (uint32_t &value, int32_t pos=0, uint32_t DefValue=CTools::m_defaultValueUINT32)
- void **GetValue** (double &value, int32_t pos=0, double DefValue=CTools::m_defaultValueDOUBLE)
- void **GetValue** (bool &value, int32_t pos=0, bool DefValue=false)
- void **GetValue** (CDate &value, int32_t pos=0)
- void **GetValue** (CDate &value, CUnit &unit, int32_t pos=0)
- void **GetValue** (CDate &value, const string &strUnit, int32_t pos=0)
- void **GetValue** (CDate &value, CUnit *unit, int32_t pos=0)
- void **GetValue** (string &value, int32_t pos=0, const string &DefValue="")
- void **GetValue** (CExpression &value, int32_t pos=0)
- void **GetValue** (CUnit &value, int32_t pos=0, const string &DefValue="count")
- void **GetValue** (uint32_t &value, string &ValueName, const KWValueListEntry *KeywordList, int32_t pos=0, uint32_t DefValue=CTools::m_defaultValueUINT32)
- void **GetValue** (bitSet32 &value, const KWValueListEntry *KeywordList, int32_t pos=0, const bitSet32 &DefValue=0)
- void **GetValue** (uint32_t &value, string &ValueName, CUIntMap &KeywordList, int32_t pos, uint32_t DefValue)
- void **GetAllValues** (CExpression &value, const string &Combine="&&")
- void **GetAllValues** (CStringList &listValues)
- void **GetAllValues** (CStringArray &listValues)

6.86.1 Detailed Description

Parameter management class.

One parameter can have 1 to n value.

This class stands for parameters

Version

1.0

6.86.2 Constructor & Destructor Documentation

6.86.2.1 brathl::CParameter::CParameter (const char * *keyword*)

Creates a new **CParameter** (p. 275) object.

Parameters

<i>keyword</i>	[in] : parameter name
----------------	-----------------------

6.86.2.2 brathl::CParameter::CParameter (const char * *keyword*, const char * *value*)

Creates a new **CParameter** (p. 275) object.

Parameters

<i>keyword</i>	[in] : parameter name
<i>value</i>	[in] : parameter value

6.86.3 Member Function Documentation

6.86.3.1 void brathl::CParameter::AddValue (const char * *value*)

Adds a value to the **CParameter** (p. 275) object.

Parameters

<i>value</i>	[in] : parameter value
--------------	------------------------

References brathl::CTools::ExpandShellVar().

Referenced by brathl::CMapParameter::Insert().

6.86.3.2 uint32_t brathl::CParameter::Count ()

Returns

the number of values.

Referenced by brathl::CFileParams::CheckCount().

6.86.3.3 void brathl::CParameter::GetValue (int32_t & *value*, int32_t *pos* = 0, int32_t *DefValue* = CTools::m_defaultValueINT32)

gets a **CParameter** (p. 275) object value at a given position If the list of values is empty or index pos is out of range a **CParameterException** (p. 278) is raised.

Parameters

<i>value</i>	[out] : parameter value
<i>pos</i>	[in] : position of the parameter 0.. n (default is 0, first value)

References BRATHL_SYNTAX_ERROR, brathl::CTools::Format(), and brathl::CTools::StrCaseCmp().

6.86.3.4 void brathl::CParameter::GetValue (char * *value*, int32_t *bufferSize*, int32_t *pos* = 0, const char * *DefValue* = " ")

gets a **CParameter** (p. 275) object value at a given position If the list of values is empty or index pos is out of range a **CParameterException** (p. 278) is raised. WARNING : if size of string value is smaller than the size of the parameter value, data will be truncated

Parameters

<i>value</i>	[out] : parameter value
<i>bufferSize</i>	[in] : size of value
<i>pos</i>	[in] : position of the parameter 0.. n (default is 0, first value)

Returns

false if one can't get the value, otherwise true

References brathl::CTools::StrCaseCmp().

6.86.3.5 bool brathl::CParameter::RemoveAllValue ()

Removes all values.

6.86.3.6 bool brathl::CParameter::RemoveValue (uint32_t i)

Removes a value at a given position. The first value is at the index 0.

Parameters

<i>i</i>	[in] : index value to remove
----------	------------------------------

6.86.3.7 void brathl::CParameter::SetAliases (const CStringMap & aliases)

Register the formulas aliases defined.

Parameters

<i>Aliases</i>	[in] : Names/values of aliases
----------------	--------------------------------

References brathl::CTools::ExpandVariables().

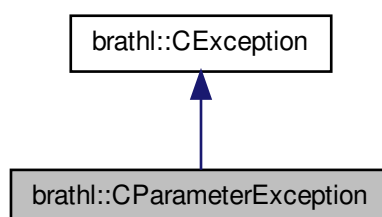
The documentation for this class was generated from the following files:

- Parameter.h
- Parameter.cpp

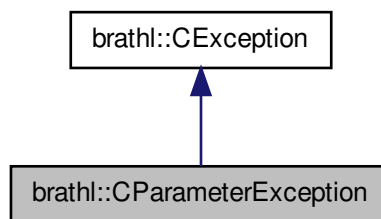
6.87 brathl::CParameterException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for brathl::CParameterException:



Collaboration diagram for brathl::CParameterException:



Public Member Functions

- **CParameterException ()**
Empty CParameterException (p. 278) ctor.
- **CParameterException (const string &message, int32_t errcode)**
- virtual const char * **TypeOf ()** const
Identification of exception (human readable)
- virtual ~**CParameterException ()** throw ()
Destructor.

Additional Inherited Members

6.87.1 Detailed Description

Parameter Exception management class.

Version

1.0

6.87.2 Constructor & Destructor Documentation

6.87.2.1 `brathl::CParameterException::CParameterException (const string & message, int32_t errcode)` `[inline]`

Creates a new **CParameterException** (p. 278) object.

Parameters

<i>message</i>	[in] : error message
<i>errcode</i>	[in] : error code

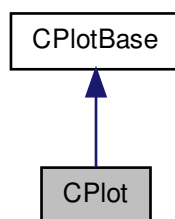
The documentation for this class was generated from the following file:

- **Exception.h**

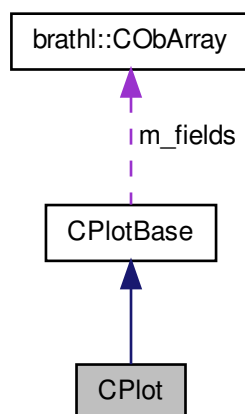
6.88 CPlot Class Reference

```
#include <Plot.h>
```

Inheritance diagram for CPlot:



Collaboration diagram for CPlot:



Public Member Functions

- **CPlot** (uint32_t groupNumber=0)
- void **GetAxisX** (**CInternalFiles** *yfx, **ExpressionValueDimensions** *dimVal, **CExpressionValue** *varX, string *varXName)
- virtual void **GetInfo** ()
- virtual **CInternalFiles** * **GetInternalFiles** (CBratObject *ob, bool withExcept=true)

Static Public Member Functions

- static **CInternalFilesYFX** * **GetInternalFilesYFX** (CBratObject *ob)

Protected Member Functions

- void **Init** ()

Additional Inherited Members

6.88.1 Detailed Description

A XY **CPlot** (p. 279) object management class

Version

1.0

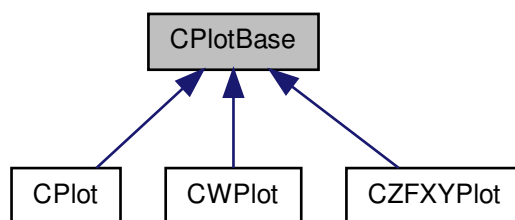
The documentation for this class was generated from the following files:

- Plot.h
- Plot.cpp

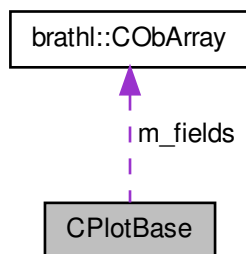
6.89 CPlotBase Class Reference

```
#include <PlotBase.h>
```

Inheritance diagram for CPlotBase:



Collaboration diagram for CPlotBase:



Public Member Functions

- **CPlotBase** (uint32_t groupNumber=0)

- **CPlotField * FindPlotField** (const wxString &fieldName, bool *withContour=NULL, bool *withSolidColor=NULL)
- void **GetAllInternalFiles** (CObArray &allInternalFiles)
- virtual void **GetForcedAxisX** (CInternalFiles *file, ExpressionValueDimensions *dimVal, CExpressionValue *varX)
- virtual void **GetForcedAxisY** (CInternalFiles *file, ExpressionValueDimensions *dimVal, CExpressionValue *varY)
- wxString **GetForcedVarXname** ()
- wxString **GetForcedVarYname** ()
- virtual void **GetInfo** ()=0
- virtual **CInternalFiles * GetInternalFiles** (CBratObject *ob, bool withExcept=true)=0
- **CPlotField * GetPlotField** (int32_t index)
- virtual void **GetVar** (const string &varName, CInternalFiles *file, ExpressionValueDimensions *dimVal, CExpressionValue *var)
- void **SetForcedVarXname** (const wxString &value)
- void **SetForcedVarYname** (const wxString &value)

Public Attributes

- **CObArray m_fields**
- wxString **m_forcedVarXName**
- wxString **m_forcedVarYName**
- uint32_t **m_groupNumber**
- CStringArray **m_nonPlotFieldNames**
- wxString **m_title**
- wxString **m_titleX**
- wxString **m_titleY**
- CUnit **m_unitX**
- bool **m_unitXConv**
- wxString **m_unitXLabel**
- CUnit **m_unitY**
- bool **m_unitYConv**
- wxString **m_unitYLabel**

6.89.1 Detailed Description

A plot object management base class

Version

1.0

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

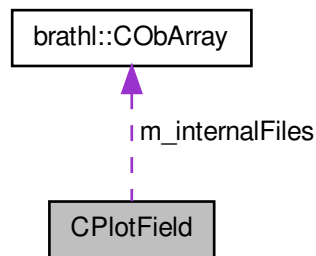
- PlotBase.h
- PlotBase.cpp

6.90 CPlotField Class Reference

```
#include <PlotField.h>
```

Inherits brathl::CBratObject.

Collaboration diagram for CPlotField:



Public Member Functions

- **CPlotField** (const wxString &name)
- **CInternalFiles * GetInternalFiles** (int32_t index)
- **CInternalFilesYFX * GetInternalFilesYFX** (int32_t index)

Static Public Member Functions

- static **CPlotField * GetPlotField** (CBratObject *ob)

Public Attributes

- **CObArray m_internalFiles**
- wxString **m_name**
- CWorldPlotProperty * **m_worldProps**
- CXYPlotProperty * **m_xyProps**
- CZFXYPlotProperty * **m_zfxyProps**

6.90.1 Detailed Description

Class to manage field and their associated internal files

Version

1.0

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

- PlotField.h
- PlotField.cpp

6.91 brathl::CProductAop Class Reference

```
#include <ProductAop.h>
```

Inherits brathl::CProduct.

Public Member Functions

- **CProductAop** ()
Empty CProductAop (p. 283) ctor.
- **CProductAop** (const string &fileName)
- **CProductAop** (const CStringList &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **InitCriteriaInfo** ()
- virtual ~**CProductAop** ()
Destructor.

Protected Member Functions

- virtual void **InitDateRef** ()

Additional Inherited Members

6.91.1 Detailed Description

Aop product management class.

Version

1.0

6.91.2 Constructor & Destructor Documentation

6.91.2.1 brathl::CProductAop::CProductAop (const string & fileName)

Creates new **CProductAop** (p. 283) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.91.2.2 brathl::CProductAop::CProductAop (const CStringList & fileNameList)

Creates new **CProductAop** (p. 283) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

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

- ProductAop.h
- ProductAop.cpp

6.92 brathl::CProductCryosat Class Reference

```
#include <ProductCryosat.h>
```

Inherits brathl::CProduct.

Public Member Functions

- **CProductCryosat** ()
Empty CProductCryosat (p. 284) ctor.
- **CProductCryosat** (const string &fileName)
- **CProductCryosat** (const CStringList &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **InitCriteriaInfo** ()
- virtual ~**CProductCryosat** ()
Destructor.

Protected Member Functions

- virtual bool **FindParentToRead** (CField *fromField, CObList *parentFieldList)
- virtual void **InitDateRef** ()
- virtual bool **IsHighResolutionField** (CField *field)
- virtual void **ProcessHighResolutionWithoutFieldCalculation** ()

Additional Inherited Members

6.92.1 Detailed Description

Cryosat product management class.

Version

1.0

6.92.2 Constructor & Destructor Documentation

6.92.2.1 brathl::CProductCryosat::CProductCryosat (const string & fileName)

Creates new **CProductCryosat** (p. 284) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.92.2.2 brathl::CProductCryosat::CProductCryosat (const CStringList & fileNameList)

Creates new **CProductCryosat** (p. 284) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

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

- ProductCryosat.h
- ProductCryosat.cpp

6.93 brathl::CProductEnvisat Class Reference

```
#include <ProductEnvisat.h>
```

Inherits brathl::CProduct.

Public Member Functions

- **CProductEnvisat** ()
Empty CProductEnvisat (p. 285) ctor.
- **CProductEnvisat** (const string &fileName)
- **CProductEnvisat** (const **CStringList** &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **InitCriteriaInfo** ()
- virtual **~CProductEnvisat** ()
Destructor.

Protected Member Functions

- virtual void **AddInternalHighResolutionFieldCalculation** ()
- void **ComputeHighResolutionFields** (**CDataSet** *dataSet)
- void **ComputeHighResolutionFields** (**CDataSet** *dataSet, double deltaLat, double deltaLon)
- virtual bool **FindParentToRead** (**CField** *fromField, **CObList** *parentFieldList)
- virtual string **GetHighResolutionLatDiffFieldName** ()
- virtual string **GetHighResolutionLonDiffFieldName** ()
- virtual bool **HasHighResolutionFieldCalculation** ()
- bool **HasHighResolutionFieldCalculationValue** (**CDataSet** *dataset)
- bool **HasHighResolutionFieldCalculationValue** (**CDataSet** *dataset, **CFieldSetArrayDbI** *fieldSetArrayDbI)
- virtual void **InitDateRef** ()
- virtual bool **IsHighResolutionField** (**CField** *field)
- bool **IsParentHighResolutionField** (**CField** *field)
- virtual void **ProcessHighResolutionWithFieldCalculation** ()
- virtual void **ProcessHighResolutionWithoutFieldCalculation** ()
- virtual void **SetHighResolutionLatDiffFieldName** (const string &value)
- virtual void **SetHighResolutionLonDiffFieldName** (const string &value)

Protected Attributes

- **CStringArray** **m_arrayTimeStampFieldName**
- string **m_highResolutionLatDiffFieldName**
- string **m_highResolutionLonDiffFieldName**
- string **m_timeStampFieldName**

Additional Inherited Members

6.93.1 Detailed Description

Envisat product management class.

Version

1.0

6.93.2 Constructor & Destructor Documentation

6.93.2.1 brathl::CProductEnvisat::CProductEnvisat (const string & *fileName*)

Creates new **CProductEnvisat** (p. 285) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.93.2.2 brathl::CProductEnvisat::CProductEnvisat (const CStringList & *fileNameList*)

Creates new **CProductEnvisat** (p. 285) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

6.93.3 Member Function Documentation

6.93.3.1 virtual string brathl::CProductEnvisat::GetHighResolutionLatDiffFieldName () [inline], [protected], [virtual]

Get the "High resolution latitude differences" field name

6.93.3.2 virtual string brathl::CProductEnvisat::GetHighResolutionLonDiffFieldName () [inline], [protected], [virtual]

Get the "High resolution longitude differences" field name

6.93.3.3 bool brathl::CProductEnvisat::IsHighResolutionField (CField * *field*) [protected], [virtual]

Determines if a field object is a 'high resolution' array data For Envisat, to be a 'high resolution' field, all conditions below have to be true :

- the field object is not an instance of **CFieldBasic** (p. 215)
- the field has one dimension and the dimension is 20.
- the field name is different from the '18 Hz latitude differences from 1 Hz' field (1) and the '18 Hz longitude differences from 1 Hz' field (1)

(1) if this field are present in the record. Note that only off-line product (product type RA2_GDR_2P and RA2_MWS_2P have these fields
- the field name contains 'hz18' or '18hz'

Parameters

<i>field</i>	[in] : field to be tested.
--------------	----------------------------

References brathl::CTools::StringToLower().

6.93.3.4 virtual void brathl::CProductEnvisat::SetHighResolutionLatDiffFieldName (const string & *value*) [inline], [protected], [virtual]

Set the "High resolution latitude differences" field name

6.93.3.5 `virtual void brathl::CProductEnvisat::SetHighResolutionLonDiffFieldName (const string & value) [inline], [protected], [virtual]`

Set the "High resolution longitude differences" field name

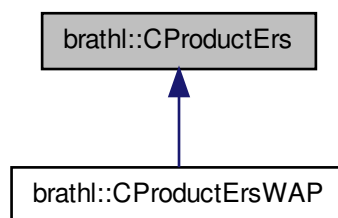
The documentation for this class was generated from the following files:

- ProductEnvisat.h
- ProductEnvisat.cpp

6.94 brathl::CProductErs Class Reference

```
#include <ProductErs.h>
```

Inheritance diagram for brathl::CProductErs:



Public Member Functions

- **CProductErs** ()
Empty CProductErs (p. 288) ctor.
- **CProductErs** (const string &fileName)
- **CProductErs** (const CStringList &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **InitCriteriaInfo** ()
- virtual **~CProductErs** ()
Destructor.

Static Public Attributes

- static const string **m_WAP** = "ALT.WAP"

Protected Member Functions

- virtual void **AddInternalHighResolutionFieldCalculation** ()
- void **ComputeHighResolutionFields** (CDataSet *dataSet, double deltaLat, double deltaLon)
- virtual void **InitDateRef** ()
- virtual bool **IsHighResolutionField** (CField *field)
- virtual void **ProcessHighResolutionWithoutFieldCalculation** ()

Protected Attributes

- string **m_timeStampMicrosecondFieldName**
- string **m_timeStampSecondFieldName**

Additional Inherited Members

6.94.1 Detailed Description

Ers product management class.

Version

1.0

6.94.2 Constructor & Destructor Documentation

6.94.2.1 brathl::CProductErs::CProductErs (const string & *fileName*)

Creates new **CProductErs** (p. 288) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.94.2.2 brathl::CProductErs::CProductErs (const CStringList & *fileNameList*)

Creates new **CProductErs** (p. 288) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

6.94.3 Member Function Documentation

6.94.3.1 bool brathl::CProductErs::IsHighResolutionField (CField * *field*) [protected], [virtual]

Determines if a field object is a 'high resolution' array data For Jason, to be a 'high resolution' field, all conditions below have to be true :

- the field object is not an instance of **CFieldBasic** (p. 215)
- the field has one dimension and the dimension is 10.

Parameters

<i>field</i>	[in] : field to be tested.
--------------	----------------------------

Reimplemented in **brathl::CProductErsWAP** (p. 291).

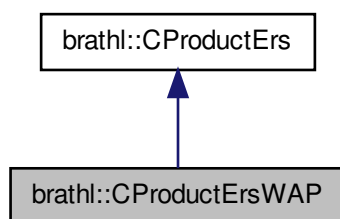
The documentation for this class was generated from the following files:

- ProductErs.h
- ProductErs.cpp

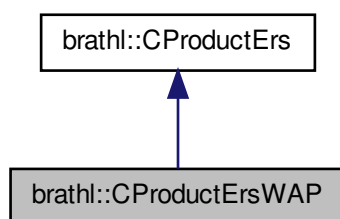
6.95 brathl::CProductErsWAP Class Reference

```
#include <ProductErsWAP.h>
```

Inheritance diagram for brathl::CProductErsWAP:



Collaboration diagram for brathl::CProductErsWAP:



Public Member Functions

- **CProductErsWAP** ()
Empty CProductErsWAP (p. 289) ctor.
- **CProductErsWAP** (const string &fileName)
- **CProductErsWAP** (const CStringList &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **InitCriteriaInfo** ()
- virtual **~CProductErsWAP** ()
Destructor.

Protected Member Functions

- virtual void **AddInternalHighResolutionFieldCalculation** ()
- void **ComputeHighResolutionFields** (CDataSet *dataSet)
- virtual bool **FindParentToRead** (CField *fromField, CObList *parentFieldList)
- virtual void **InitDateRef** ()
- virtual bool **IsDirectHighResolutionField** (CField *field)
- virtual bool **IsHighResolutionField** (CField *field)
- virtual void **ProcessHighResolutionWithoutFieldCalculation** ()

Protected Attributes

- string **m_timeStampDayFieldName**
- string **m_timeStampMicrosecondFieldName**
- string **m_timeStampMillisecondFieldName**

Additional Inherited Members

6.95.1 Detailed Description

Ers product management class.

Version

1.0

6.95.2 Constructor & Destructor Documentation

6.95.2.1 brathl::CProductErsWAP::CProductErsWAP (const string & *fileName*)

Creates new **CProductErsWAP** (p. 289) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.95.2.2 brathl::CProductErsWAP::CProductErsWAP (const CStringList & *fileNameList*)

Creates new **CProductErsWAP** (p. 289) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

6.95.3 Member Function Documentation

6.95.3.1 bool brathl::CProductErsWAP::IsHighResolutionField (CField * *field*) [protected], [virtual]

Determines if a field object is a 'high resolution' array data For Jason, to be a 'high resolution' field, all conditions below have to be true :

- the field object is not an instance of **CFieldBasic** (p. 215)
- the field has one dimension and the dimension is 10.

Parameters

<i>field</i>	[in] : field to be tested.
--------------	----------------------------

Reimplemented from **brathl::CProductErs** (p. 289).

References BRATHL_INCONSISTENCY_ERROR, BRATHL_UNIMPLEMENT_ERROR, and brathl::CTools::Format().

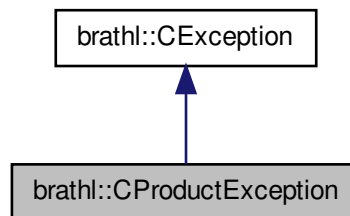
The documentation for this class was generated from the following files:

- ProductErsWAP.h
- ProductErsWAP.cpp

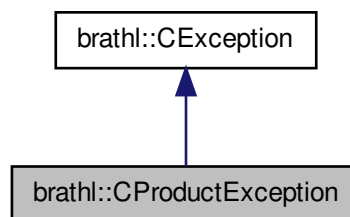
6.96 bratl::CProductException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for bratl::CProductException:



Collaboration diagram for bratl::CProductException:



Public Member Functions

- **CProductException** ()
Empty CProductException (p. 292) ctor.
- **CProductException** (const string &message, int32_t errcode)
- **CProductException** (const string &message, const string &fileName, int32_t errcode)
- **CProductException** (const string &message, const string &fileName, const string &productClass, const string &productType, int32_t errcode)
- virtual const char * **TypeOf** () const
Identification of exception (human readable)
- virtual ~**CProductException** () throw ()
Destructor.

Additional Inherited Members

6.96.1 Detailed Description

Product Exception management class.

Version

1.0

6.96.2 Constructor & Destructor Documentation

6.96.2.1 brathl::CProductException::CProductException (const string & *message*, int32_t *errcode*) [inline]

Creates a new **CProductException** (p. 292) object.

Parameters

<i>message</i>	[in] : error message
<i>errcode</i>	[in] : error code

6.96.2.2 brathl::CProductException::CProductException (const string & *message*, const string & *fileName*, int32_t *errcode*)

Creates a new **CFileException** (p. 243) object.

Parameters

<i>message</i>	[in] : error message
<i>fileName</i>	[in] : file name in error
<i>errcode</i>	[in] : error code

6.96.2.3 brathl::CProductException::CProductException (const string & *message*, const string & *fileName*, const string & *productClass*, const string & *productType*, int32_t *errcode*)

Creates a new CProductException object.

Parameters

<i>message</i>	[in] : error message
<i>fileName</i>	[in] : product file name
<i>productClass</i>	[in] : product class
<i>productType</i>	[in] : product type
<i>errcode</i>	[in] : error code

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

- **Exception.h**
- Exception.cpp

6.97 brathl::CProductGfo Class Reference

```
#include <ProductGfo.h>
```

Inherits brathl::CProduct.

Public Member Functions

- **CProductGfo** ()
Empty **CProductGfo** (p. 293) ctor.
- **CProductGfo** (const string &fileName)
- **CProductGfo** (const **CStringList** &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)

Dump fonction.

- virtual void **InitCriteriaInfo** ()
- virtual \sim **CProductGfo** ()

Destructor.

Protected Member Functions

- virtual void **AddInternalHighResolutionFieldCalculation** ()
- void **ComputeHighResolutionFields** (**CDataSet** *dataSet, double deltaLat, double deltaLon)
- virtual void **InitDateRef** ()
- virtual bool **IsHighResolutionField** (**CField** *field)
- virtual void **ProcessHighResolutionWithoutFieldCalculation** ()

Protected Attributes

- string **m_timeStampMicrosecondFieldName**
- string **m_timeStampSecondFieldName**

Additional Inherited Members

6.97.1 Detailed Description

Ers product management class.

Version

1.0

6.97.2 Constructor & Destructor Documentation

6.97.2.1 bratl::CProductGfo::CProductGfo (const string & fileName)

Creates new **CProductGfo** (p. 293) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.97.2.2 bratl::CProductGfo::CProductGfo (const CStringList & fileNameList)

Creates new **CProductGfo** (p. 293) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

6.97.3 Member Function Documentation

6.97.3.1 bool bratl::CProductGfo::IsHighResolutionField (CField * field) [protected], [virtual]

Determines if a field object is a 'high resolution' array data For Jason, to be a 'high resolution' field, all conditions below have to be true :

- the field object is not an instance of **CFieldBasic** (p. 215)

- the field has one dimension and the dimension is 10.

Parameters

<i>field</i>	[in] : field to be tested.
--------------	----------------------------

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

- ProductGfo.h
- ProductGfo.cpp

6.98 brathl::CProductJason Class Reference

```
#include <ProductJason.h>
```

Inherits brathl::CProduct.

Public Member Functions

- **CProductJason** ()
Empty CProductJason (p. 295) ctor.
- **CProductJason** (const string &fileName)
- **CProductJason** (const CStringList &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **InitCriteriaInfo** ()
- virtual **~CProductJason** ()
Destructor.

Protected Member Functions

- virtual void **AddInternalHighResolutionFieldCalculation** ()
- void **ComputeHighResolutionFields** (CDataSet *dataSet, double deltaLat, double deltaLon)
- virtual void **InitDateRef** ()
- virtual bool **IsHighResolutionField** (CField *field)
- virtual void **ProcessHighResolutionWithoutFieldCalculation** ()

Protected Attributes

- string **m_timeStampDayFieldName**
- string **m_timeStampMicrosecondFieldName**
- string **m_timeStampSecondFieldName**

Additional Inherited Members

6.98.1 Detailed Description

Jason product management class.

Version

1.0

6.98.2 Constructor & Destructor Documentation

6.98.2.1 brathl::CProductJason::CProductJason (const string & *fileName*)

Creates new **CProductJason** (p. 295) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.98.2.2 brathl::CProductJason::CProductJason (const CStringList & *fileNameList*)

Creates new **CProductJason** (p. 295) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

6.98.3 Member Function Documentation

6.98.3.1 bool brathl::CProductJason::IsHighResolutionField (CField * *field*) [protected], [virtual]

Determines if a field object is a 'high resolution' array data For Jason, to be a 'high resolution' field, all conditions below have to be true :

- the field object is not an instance of **CFieldBasic** (p. 215)
- the field has one dimension and the dimension is 20.

Parameters

<i>field</i>	[in] : field to be tested.
--------------	----------------------------

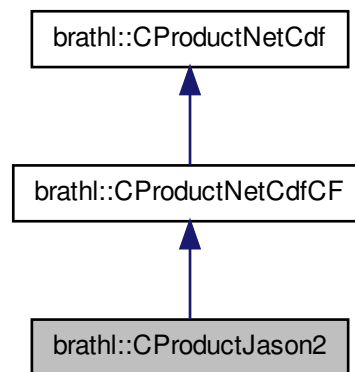
The documentation for this class was generated from the following files:

- ProductJason.h
- ProductJason.cpp

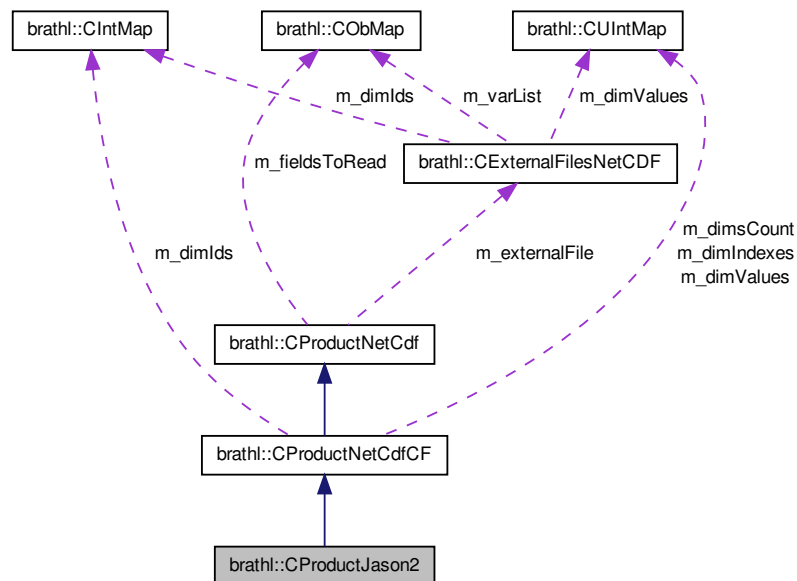
6.99 brathl::CProductJason2 Class Reference

```
#include <ProductJason2.h>
```

Inheritance diagram for brathl::CProductJason2:



Collaboration diagram for brathl::CProductJason2:



Public Member Functions

- **CProductJason2** ()
CIntMap (p. 257) ctor.
- **CProductJason2** (const string &fileName)
- **CProductJason2** (const **CStringList** &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump function.

- virtual bool **HasCriterialInfo** ()
- virtual void **InitCriterialInfo** ()
- virtual void **InitDateRef** ()

Protected Member Functions

- void **Init** ()

Additional Inherited Members

6.99.1 Detailed Description

Mapping products management class.

Version

1.0

6.99.2 Constructor & Destructor Documentation

6.99.2.1 CProductJason2::CProductJason2 (const string & *fileName*)

Creates new **CProductNetCdf** (p. 300) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.99.2.2 CProductJason2::CProductJason2 (const CStringList & *fileNameList*)

Creates new **CProductNetCdf** (p. 300) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

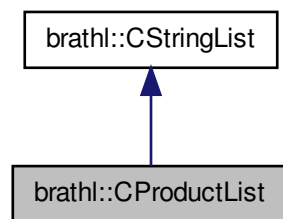
The documentation for this class was generated from the following files:

- ProductJason2.h
- ProductJason2.cpp

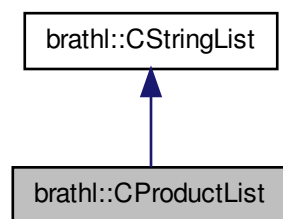
6.100 brathl::CProductList Class Reference

```
#include <Product.h>
```

Inheritance diagram for bratl::CProductList:



Collaboration diagram for bratl::CProductList:



Public Member Functions

- bool **CheckFiles** (bool onlyFirstFile=false)
- bool **CheckFilesNetCdf** ()
- **CProductList** ()
 - Empty CProductList (p. 298) ctor.*
- **CProductList** (const **CProductList** &p)
- **CProductList** (const string &fileName)
- **CProductList** (const **CStringList** &fileNameList)
- **CProductList** (const CStringArray &fileNameArray)
- virtual void **Dump** (ostream &fOut=cerr)
 - Dump fonction.*
- const string **GetMessage** ()
- bool **IsATP** ()
- bool **IsGenericNetCdf** ()
- bool **IsHdfOrNetcdfCodaFormat** ()
- bool **IsJason2** ()
- bool **IsNetCdfCFProduct** ()
- bool **IsNetCdfOrNetCdfCFProduct** ()
- bool **IsNetCdfProduct** ()
- bool **IsSameProduct** (const string &productClass, const string &productType)

- bool **IsYFX** ()
- bool **IsZFX** ()
- const **CProductList** & **operator=** (const **CProductList** &lst)
- void **Set** (const **CProductList** &lst)
- virtual ~**CProductList** ()

Destructor.

Static Public Member Functions

- static bool **IsHdfOrNetcdfCodaFormat** (coda_format format)

Public Attributes

- string **m_message**
- string **m_productClass**
- coda_format **m_productFormat**
- string **m_productType**

Protected Member Functions

- bool **CheckFileList** ()

6.100.1 Detailed Description

Product file list management class.

Version

1.0

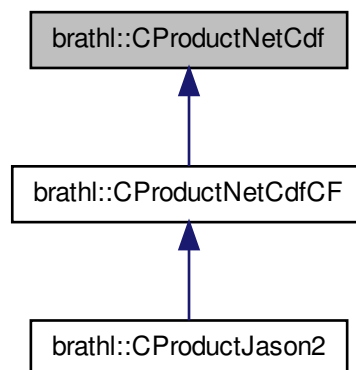
The documentation for this class was generated from the following files:

- Product.h
- Product.cpp

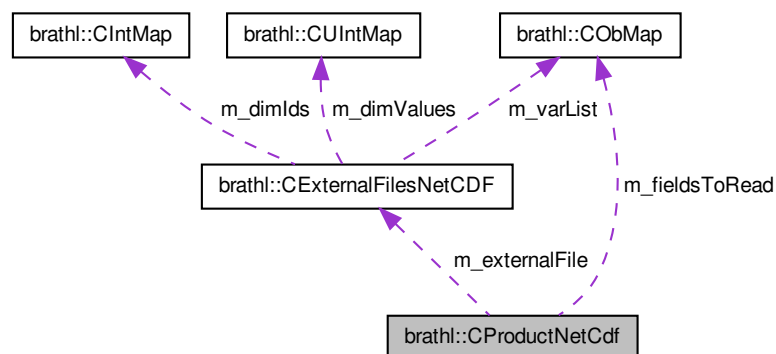
6.101 brathl::CProductNetCdf Class Reference

```
#include <ProductNetCdf.h>
```


Inheritance diagram for bratl::CProductNetCdf:



Collaboration diagram for bratl::CProductNetCdf:



Public Member Functions

- void **AddDimsToReadOneByOne** (const CStringArray &value)
- virtual void **AddOffset** (double value, CField *field=NULL)
- virtual void **ApplyCriteria** (CStringList &filteredFileList, const string &logFileName="")
- virtual bool **ApplyCriteriaCycle** (CCriterialInfo *criterialInfo)
- virtual bool **ApplyCriteriaDatetime** (CCriterialInfo *criterialInfo)
- virtual bool **ApplyCriteriaLatLon** (CCriterialInfo *criterialInfo)
- virtual bool **ApplyCriteriaPass** (CCriterialInfo *criterialInfo)
- virtual bool **ApplyCriteriaPassInt** (CCriterialInfo *criterialInfo)
- virtual bool **ApplyCriteriaPassString** (CCriterialInfo *criterialInfo)
- virtual void **CheckFileOpened** ()
- virtual CProduct * **Clone** ()
- virtual bool **Close** ()

- **CProductNetCdf** ()
Empty CProductNetCdf (p. 300) ctor.
- **CProductNetCdf** (const string &fileName)
- **CProductNetCdf** (const CStringList &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- const CStringArray * **GetAxisDims** ()
- CStringArray * **GetComplementDims** ()
- virtual bool **GetDateMinMax** (CDatePeriod &datePeriodMinMax)
- CStringArray * **GetDimsToReadOneByOne** ()
- **CExternalFilesNetCDF** * **GetExternalFile** ()
- virtual bool **GetForceReadDataOneByOne** ()
- virtual bool **GetLatLonMinMax** (CLatLonRect &latlonRectMinMax)
- void **GetNetCdfDimensions** (const vector< CExpression > &expressions, CStringArray &commonDimNames)
- void **GetNetCdfDimensions** (const CExpression &expr, CStringArray &commonDimNames)
- void **GetNetCdfDimensions** (const CStringArray &fields, CStringArray &commonDimNames)
- void **GetNetCdfDimensions** (const vector< CExpression > &expressions, CStringArray &commonDimNames, const string &recordName)
- void **GetNetCdfDimensions** (const CExpression &expr, CStringArray &commonDimNames, const string &recordName)
- void **GetNetCdfDimensions** (const CStringArray &fields, CStringArray &commonDimNames, const string &recordName)
- void **GetNetCdfDimensionsWithoutAlgo** (const vector< CExpression > &expressions, CStringArray &commonDimNames, const string &recordName)
- void **GetNetCdfDimensionsWithoutAlgo** (const CExpression &expr, CStringArray &commonDimNames, const string &recordName)
- virtual int32_t **GetNumberOfRecords** (const string &dataSetName)
- virtual int32_t **GetNumberOfRecords** ()
- virtual void **GetRecords** (CStringArray &array)
- virtual bool **HasCriteriaInfo** ()
- virtual void **InitCriteriaInfo** ()
- void **InitDataset** ()
- virtual void **InitDateRef** ()
- void **InitLatLonFieldName** ()
- bool **IsApplyNetcdfProductInitialisation** ()
- bool **IsLatField** (CFieldNetCdf *field)
- bool **IsLonField** (CFieldNetCdf *field)
- virtual bool **IsOpened** ()
- virtual bool **IsOpened** (const string &fileName)
- void **MustBeOpened** ()
- virtual void **NetCdfProductInitialization** (CProduct *from)
- virtual bool **NextRecord** ()
- virtual bool **Open** (const string &fileName, const string &dataSetName, CStringList &listFieldToRead)
- virtual bool **Open** (const string &fileName, const string &dataSetName)
- virtual bool **Open** (const string &fileName)
- virtual bool **PrevRecord** ()
- virtual void **ReadBratRecord** (int32_t iRecord)
- **CFieldNetCdf** * **ReadDateCriteriaValue** (CFieldInfo &fieldInfo, CDate &date, bool wantMin=true)
- **CFieldNetCdf** * **ReadDoubleCriteriaValue** (CFieldInfo &fieldInfo, double &value, bool wantMin=true)
- virtual void **Rewind** ()
- void **SetApplyNetcdfProductInitialisation** (bool value)
- void **SetAxisDims** (const CStringArray &value)
- void **SetComplementDims** (const CStringArray &value)
- void **SetDimsToReadOneByOne** (const CStringArray &value)

- virtual void **SetForceReadDataOneByOne** (bool value)
- virtual void **SetOffset** (double value)
- virtual **~CProductNetCdf** ()

Destructor.

Static Public Member Functions

- static **CProductNetCdf * GetProductNetCdf** (CBratObject *ob, bool withExcept=true)
- static bool **IsProductNetCdf** (CBratObject *ob)

Static Public Attributes

- static const string **m_virtualRecordName** = "data"

Protected Member Functions

- virtual void **CreateFieldSets** ()
- void **DeleteExternalFile** ()
- void **DeleteFieldsToReadMap** ()
- virtual void **FillDescription** ()
- **CFieldNetCdf * FindCycleField** ()
- **CFieldNetCdf * FindLatField** ()
- **CFieldNetCdf * FindLonField** ()
- **CFieldNetCdf * FindPassField** ()
- **CFieldNetCdf * FindTimeField** ()
- void **Init** ()
- virtual void **InitInternalFieldName** (const string &dataSetName, **CStringList** &listField, bool convertDate=false)
- virtual void **InitInternalFieldName** (**CStringList** &listField, bool convertDate=false)
- virtual void **LoadFieldsInfo** ()
- virtual string **MakeInternalFieldName** (const string &dataSetName, const string &field)
- virtual string **MakeInternalFieldName** (const string &field)
- virtual bool **Open** ()
- virtual **CFieldNetCdf * Read** (CFieldInfo &fieldInfo, double &value, bool wantMin=true)
- virtual void **Read** (CFieldInfo &fieldInfo, string &value)
- virtual void **Read** (**CFieldNetCdf *field**, double &value)
- virtual void **Read** (**CFieldNetCdf *field**, **CDoubleArray** &vect)
- virtual void **Read** (**CFieldNetCdf *field**, **CExpressionValue** &value)
- virtual void **ReadAll** (**CFieldNetCdf *field**)
- virtual void **ReadAll** (**CFieldNetCdf *field**, **CExpressionValue** &value)
- virtual void **ReadBratFieldRecord** (const string &key)
- virtual void **ReadBratFieldRecord** (CField::CListField::iterator it)
- virtual void **RewindEnd** ()
- virtual void **RewindInit** ()
- virtual void **RewindProcess** ()

Protected Attributes

- bool **m_applyNetcdfProductInitialisation**
- CStringArray **m_axisDims**
- CStringArray **m_complementDims**
- CStringArray **m_dimsToReadOneByOne**
- **CExternalFilesNetCDF * m_externalFile**
- **CObMap * m_fieldsToRead**
- bool **m_forceReadDataOneByOne**

6.101.1 Detailed Description

Netcdf product management class.

Version

1.0

6.101.2 Constructor & Destructor Documentation

6.101.2.1 brathl::CProductNetCdf::CProductNetCdf (const string & *fileName*)

Creates new **CProductNetCdf** (p. 300) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.101.2.2 brathl::CProductNetCdf::CProductNetCdf (const CStringList & *fileNameList*)

Creates new **CProductNetCdf** (p. 300) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

6.101.3 Member Data Documentation

6.101.3.1 CObMap* brathl::CProductNetCdf::m_fieldsToRead [protected]

Map of the fields to read (key : var name → **CFieldNetCdf** (p. 218) object) NB : **CFieldNetCdf** (p. 218) objects stored in this map have not to be delete (they are not a copy !!!)

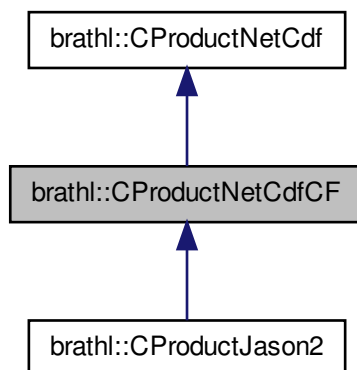
The documentation for this class was generated from the following files:

- ProductNetCdf.h
- ProductNetCdf.cpp

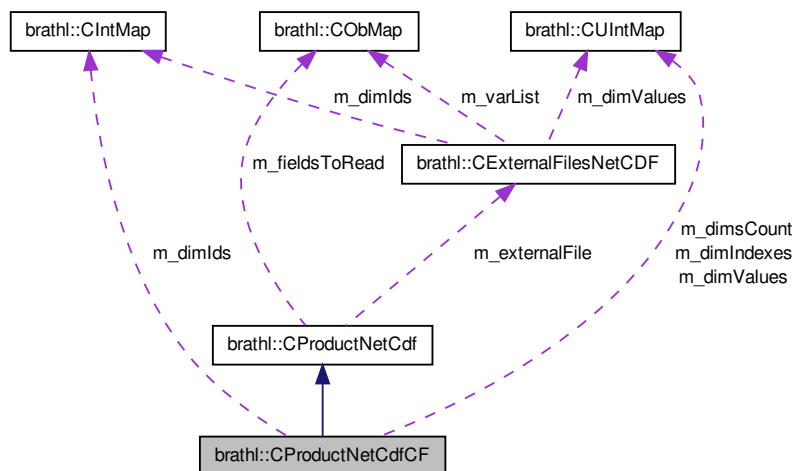
6.102 brathl::CProductNetCdfCF Class Reference

```
#include <ProductNetCdfCF.h>
```

Inheritance diagram for brathl::CProductNetCdfCF:



Collaboration diagram for brathl::CProductNetCdfCF:



Public Member Functions

- virtual `CProduct *` **Clone** ()
- **CProductNetCdfCF** ()
Empty *CProductNetCdf* (p. 300) ctor.
- **CProductNetCdfCF** (const string &fileName)
- **CProductNetCdfCF** (const **CStringList** &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual int32_t **GetNumberOfRecords** (const string &dataSetName)
- virtual int32_t **GetNumberOfRecords** ()

- virtual bool **NextRecord** ()
- virtual bool **PrevRecord** ()
- virtual void **Rewind** ()
- virtual **~CProductNetCdfCF** ()

Destructor.

Static Public Member Functions

- static **CProductNetCdfCF * GetProductNetCdfCF** (CBratObject *ob, bool withExcept=true)
- static bool **IsProductNetCdfCF** (CBratObject *ob)

Protected Member Functions

- void **AdjustIndexesFromField** (CFieldNetCdf *field, bool next=true)
- void **AdjustIndexesToMin** (bool next=true)
- void **AdjustIndexesToMin** (CFieldNetCdf *field, bool next=true)
- bool **CheckEOF** ()
- void **Init** ()
- void **InitDimIndexes** (uint32_t value)
- virtual void **InitDimsIndexToMax** ()
- bool **IsAtBeginning** ()
- bool **NextFieldIndex** ()
- bool **PrevFieldIndex** ()
- virtual void **RewindEnd** ()
- virtual void **RewindInit** ()
- virtual void **RewindProcess** ()
- void **SetFieldIndex** ()
- void **SetFieldIndex** (CFieldNetCdf *field)

Protected Attributes

- bool **m_atBeginning**
- CIntMap **m_dimIds**
- CUIntMap **m_dimIndexes**
- CUIntMap **m_dimsCount**
- CUIntMap **m_dimValues**

Additional Inherited Members

6.102.1 Detailed Description

Netcdf product management class.

Version

1.0

6.102.2 Constructor & Destructor Documentation

6.102.2.1 brathl::CProductNetCdfCF::CProductNetCdfCF (const string & fileName)

Creates new **CProductNetCdf** (p. 300) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.102.2.2 brathl::CProductNetCdfCF::CProductNetCdfCF (const CStringList & fileNameList)

Creates new **CProductNetCdf** (p. 300) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

6.102.3 Member Data Documentation

6.102.3.1 bool brathl::CProductNetCdfCF::m_atBeginning [protected]

'At beginning" flag

Referenced by Dump().

6.102.3.2 CIntMap brathl::CProductNetCdfCF::m_dimIds [protected]

Map of the dimension's ids of the read fields (key : dim name → dim ids)

Referenced by Dump().

6.102.3.3 CUIntMap brathl::CProductNetCdfCF::m_dimsCount [protected]

Map of the dimension's ranges of the read fields (key : dim name → dim range)Array of the dimension count for reading (key : dim name → count)

Referenced by Dump().

6.102.3.4 CUIntMap brathl::CProductNetCdfCF::m_dimValues [protected]

Map of the dimension's values of the read fields (key : dim name → dim value)

Referenced by Dump().

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

- ProductNetCdfCF.h
- ProductNetCdfCF.cpp

6.103 brathl::CProductPodaac Class Reference

```
#include <ProductPodaac.h>
```

Inherits brathl::CProduct.

Public Member Functions

- **CProductPodaac** ()
Empty CProductPodaac (p. 307) ctor.
- **CProductPodaac** (const string &fileName)
- **CProductPodaac** (const CStringList &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual string **GetLabel** ()
- virtual void **InitCriteriaInfo** ()

- virtual `~CProductPodaac ()`
Destructor.

Static Public Attributes

- static const string `m_J1SSHA_ATG_FILE` = "J1SSHA_ATG_FILE"
- static const string `m_J1SSHA_PASS_FILE` = "J1SSHA_PASS_FILE"
- static const string `m_TPSSHA_ATG_FILE` = "TPSSHA_ATG_FILE"
- static const string `m_TPSSHA_PASS_FILE` = "TPSSHA_PASS_FILE"

Protected Member Functions

- virtual void `InitDateRef ()`

Additional Inherited Members

6.103.1 Detailed Description

Ers product management class.

Version

1.0

6.103.2 Constructor & Destructor Documentation

6.103.2.1 brathl::CProductPodaac::CProductPodaac (const string & *fileName*)

Creates new **CProductPodaac** (p. 307) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.103.2.2 brathl::CProductPodaac::CProductPodaac (const CStringList & *fileNameList*)

Creates new **CProductPodaac** (p. 307) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

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

- ProductPodaac.h
- ProductPodaac.cpp

6.104 brathl::CProductRads Class Reference

```
#include <ProductRads.h>
```

Inherits brathl::CProduct.

Public Member Functions

- **CProductRads** ()
Empty CProductRads (p. 308) ctor.
- **CProductRads** (const string &fileName)
- **CProductRads** (const CStringList &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **InitCriteriaInfo** ()
- virtual **~CProductRads** ()
Destructor.

Protected Member Functions

- virtual void **InitDateRef** ()

Additional Inherited Members

6.104.1 Detailed Description

RADS product management class.

Version

1.0

6.104.2 Constructor & Destructor Documentation

6.104.2.1 brathl::CProductRads::CProductRads (const string & fileName)

Creates new **CProductRads** (p. 308) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.104.2.2 brathl::CProductRads::CProductRads (const CStringList & fileNameList)

Creates new **CProductRads** (p. 308) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

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

- ProductRads.h
- ProductRads.cpp

6.105 brathl::CProductRiverLake Class Reference

```
#include <ProductRiverLake.h>
```

Inherits brathl::CProduct.

Public Member Functions

- **CProductRiverLake** ()
Empty CProductRiverLake (p. 309) ctor.
- **CProductRiverLake** (const string &fileName)
- **CProductRiverLake** (const **CStringList** &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual void **InitCriteriaInfo** ()
- virtual ~**CProductRiverLake** ()
Destructor.

Protected Member Functions

- virtual void **InitDateRef** ()

Additional Inherited Members

6.105.1 Detailed Description

River & Lake product management class.

Version

1.0

6.105.2 Constructor & Destructor Documentation

6.105.2.1 brathl::CProductRiverLake::CProductRiverLake (const string & fileName)

Creates new **CProductRiverLake** (p. 309) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.105.2.2 brathl::CProductRiverLake::CProductRiverLake (const CStringList & fileNameList)

Creates new **CProductRiverLake** (p. 309) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

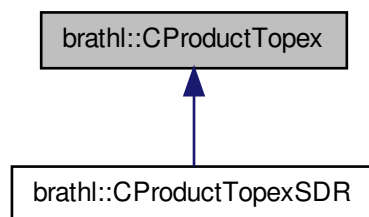
The documentation for this class was generated from the following files:

- ProductRiverLake.h
- ProductRiverLake.cpp

6.106 brathl::CProductTopex Class Reference

```
#include <ProductTopex.h>
```

Inheritance diagram for bratl::CProductTopex:



Public Member Functions

- **CProductTopex** ()
Empty CProductTopex (p. 310) ctor.
- **CProductTopex** (const string &fileName)
- **CProductTopex** (const CStringList &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual string **GetLabel** ()
- virtual void **InitCriteriaInfo** ()
- virtual **~CProductTopex** ()
Destructor.

Static Public Attributes

- static const int32_t **m_ALTIMETER_POSEIDON** = 0
- static const int32_t **m_ALTIMETER_TOPEX** = 1
- static const string **m_PASS_FILE** = "MGDR_pass_file"
- static const string **m_SDR_PASS_FILE** = "SDR_pass_file"
- static const string **m_TOPEX_POSEIDON_HEADER** = "header"
- static const string **m_XNG_FILE** = "MGDR_crossover_point_file"

Protected Member Functions

- virtual void **AddInternalHighResolutionFieldCalculation** ()
- void **ComputeHighResolutionFields** (CDataSet *dataSet, double deltaLat, double deltaLon)
- virtual void **InitDateRef** ()
- virtual bool **IsHighResolutionField** (CField *field)
- virtual void **ProcessHighResolutionWithoutFieldCalculation** ()
- virtual void **SetDeltaTimeHighResolution** (int32_t altimeterIndicator)

Protected Attributes

- string **m_altimeterIndicatorFieldName**
- string **m_timeStampDayFieldName**
- string **m_timeStampMicrosecondFieldName**
- string **m_timeStampMillisecondFieldName**

Additional Inherited Members

6.106.1 Detailed Description

Topex/Poseidon product management class.

Version

1.0

6.106.2 Constructor & Destructor Documentation

6.106.2.1 bratl::CProductTopex::CProductTopex (const string & fileName)

Creates new **CProductTopex** (p. 310) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.106.2.2 bratl::CProductTopex::CProductTopex (const CStringList & fileNameList)

Creates new **CProductTopex** (p. 310) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

6.106.3 Member Function Documentation

6.106.3.1 bool bratl::CProductTopex::IsHighResolutionField (CField * field) [protected], [virtual]

Determines if a field object is a 'high resolution' array data For Topex/Poseidon, to be a 'high resolution' field, all conditions below have to be true :

- the field object is not an instance of **CFieldBasic** (p. 215)
- the field has one dimension and the dimension is 10.

Parameters

<i>field</i>	[in] : field to be tested.
--------------	----------------------------

Reimplemented in **bratl::CProductTopexSDR** (p. 314).

6.106.4 Member Data Documentation

6.106.4.1 const int32_t bratl::CProductTopex::m_ALTIMETER_POSEIDON = 0 [static]

Altimeter Indicator. This element is computed for TOPEX and POSEIDON data. It indicates which altimeter is on at the time of the measurement. Value Definition: 0 = POSEIDON on, 1 = TOPEX on

6.106.4.2 string bratl::CProductTopex::m_altimeterIndicatorFieldName [protected]

Altimeter Indicator. This element is computed for TOPEX and POSEIDON data. It indicates which altimeter is on at the time of the measurement. Value Definition: 0 = POSEIDON on, 1 = TOPEX on

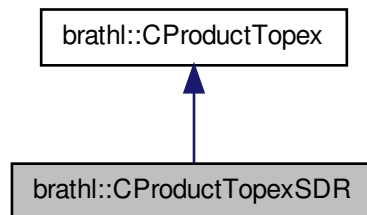
The documentation for this class was generated from the following files:

- ProductTopex.h
- ProductTopex.cpp

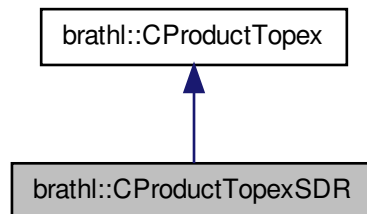
6.107 bratl::CProductTopexSDR Class Reference

```
#include <ProductTopexSDR.h>
```

Inheritance diagram for bratl::CProductTopexSDR:



Collaboration diagram for bratl::CProductTopexSDR:



Public Member Functions

- **CProductTopexSDR** ()
*Empty **CProductTopexSDR** (p. 313) ctor.*
- **CProductTopexSDR** (const string &fileName)
- **CProductTopexSDR** (const **CStringList** &fileNameList)
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual string **GetLabel** ()
- virtual **~CProductTopexSDR** ()
Destructor.

Protected Member Functions

- virtual void **CheckConsistencyHighResolutionField** (CFieldSetArrayDbI *fieldSetArrayDbI)
- void **ComputeHighResolutionFields** (CDataSet *dataSet, double deltaLat, double deltaLon)
- virtual bool **IsHighResolutionField** (CField *field)
- virtual void **ProcessHighResolutionWithoutFieldCalculation** ()
- virtual void **PutFlatHighResolution** (CDataSet *dataSet, CFieldSetArrayDbI *fieldSetArrayDbI)
- virtual void **SetHighResolution** (CField *field)

Protected Attributes

- uint32_t **m_highRateNumHighResolutionMeasure**
- uint32_t **m_lowRateNumHighResolutionMeasure**

Additional Inherited Members

6.107.1 Detailed Description

Topex/Poseidon SDR product management class.

Version

1.0

6.107.2 Constructor & Destructor Documentation

6.107.2.1 brathl::CProductTopexSDR::CProductTopexSDR (const string & *fileName*)

Creates new **CProductTopexSDR** (p. 313) object

Parameters

<i>fileName</i>	[in] : file name to be connected
-----------------	----------------------------------

6.107.2.2 brathl::CProductTopexSDR::CProductTopexSDR (const CStringList & *fileNameList*)

Creates new **CProductTopexSDR** (p. 313) object

Parameters

<i>fileNameList</i>	[in] : list of file to be connected
---------------------	-------------------------------------

6.107.3 Member Function Documentation

6.107.3.1 bool brathl::CProductTopexSDR::IsHighResolutionField (CField * *field*) [protected], [virtual]

Determines if a field object is a 'high resolution' array data For Topex/Poseidon, to be a 'high resolution' field, all conditions below have to be true :

- **CProductTopex** (p. 310) rules (see **CProductTopex::IsHighResolutionField** (p. 312))
- the field has two dimensions and the first dimension is 10 or 5.

Parameters

<i>field</i>	[in] : field to be tested.
--------------	----------------------------

Reimplemented from **brathl::CProductTopex** (p. 312).

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

- ProductTopexSDR.h
- ProductTopexSDR.cpp

6.108 brathl::CPtrMap Class Reference

```
#include <List.h>
```

Public Member Functions

- **CPtrMap** (bool bDelete=true)
CPtrMap (p. 315) *ctor.*
- virtual void **Dump** (ostream &fOut=cerr) const
Dump function.
- virtual bool **Erase** (CPtrMap::iterator it)
- virtual bool **Erase** (const string &key)
- virtual void * **Exists** (const string &key) const
- virtual void * **Insert** (const string &key, void *ptr, bool withExcept=true)
- virtual void **Insert** (const **CPtrMap** &ptrMap, bool withExcept=true)
- virtual void * **operator[]** (const string &key)
- virtual void **RemoveAll** ()
- virtual ~**CPtrMap** ()
CPtrMap (p. 315) *dtor.*

Protected Attributes

- bool **m_bDelete**

6.108.1 Detailed Description

a set of pointer management classes.

Version

1.0

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

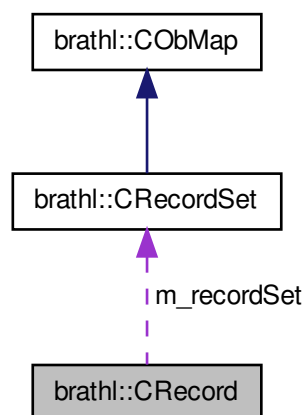
- List.h
- List.cpp

6.109 brathl::CRecord Class Reference

```
#include <Field.h>
```

Inherits brathl::CBratObject.

Collaboration diagram for bratl::CRecord:



Public Member Functions

- **CRecord** (**CRecordSet** *recordSet=NULL)
Ctor.
- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- const string & **GetName** ()
- **CRecordSet** * **GetRecordSet** ()
- virtual ~**CRecord** ()
Dtor.

Protected Attributes

- **CRecordSet** * **m_recordSet**

6.109.1 Detailed Description

a set of record management classes.

Version

1.0

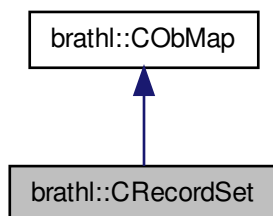
The documentation for this class was generated from the following files:

- Field.h
- Field.cpp

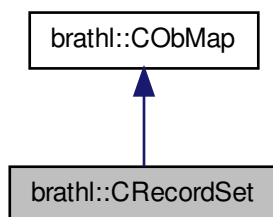
6.110 brathl::CRecordSet Class Reference

```
#include <Field.h>
```

Inheritance diagram for brathl::CRecordSet:



Collaboration diagram for brathl::CRecordSet:



Public Member Functions

- **CRecordSet** (const string &name="", bool bDelete=true)
Ctor.
- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- void **ExecuteExpression** (CExpression &expr, const string &recordName, **CExpressionValue** &exprValue, CProduct *product=NULL)
- **CFieldSet** * **ExistsFieldSet** (const string &key)
- **CField** * **GetField** (CRecordSet::iterator it)
- **CFieldSet** * **GetFieldSet** (CRecordSet::iterator it)
- **CFieldSet** * **GetFieldSet** (const string &dataSetName, const string &fieldName)
- bool **IsFieldHasToBeExpanded** (CRecordSet::iterator it, const **CStringList** &listFieldExpandArray)
- bool **IsFieldHasToBeExpanded** (**CFieldSet** *fieldSet, const **CStringList** &listFieldExpandArray)
- virtual ~**CRecordSet** ()
Dtor.

Public Attributes

- string **m_name**

Additional Inherited Members

6.110.1 Detailed Description

a set of record fields value management classes.

Version

1.0

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

- Field.h
- Field.cpp

6.111 brathl::CRegisteredPass Class Reference

```
#include <ExternalFilesATP.h>
```

Inherits brathl::CBratObject.

Public Member Functions

- **CRegisteredPass** (**CRegisteredPass** &p)
- const **CRegisteredPass** & **operator=** (**CRegisteredPass** &p)
- void **Set** (**CRegisteredPass** &p)

Public Attributes

- double **m_beginDate**
- uint32_t **m_cycle**
- uint32_t **m_cycleIndex**
- uint32_t **m_nbData**
- uint32_t **m_pass**
- uint32_t **m_startPoint**

6.111.1 Detailed Description

External files access.

Version

1.0

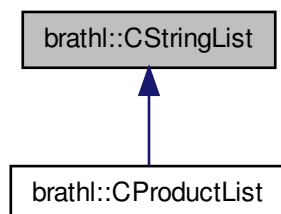
The documentation for this class was generated from the following file:

- ExternalFilesATP.h

6.112 bratl::CStringList Class Reference

```
#include <List.h>
```

Inheritance diagram for bratl::CStringList:



Public Member Functions

- virtual bool **Complement** (const **CStringList** &array, **CStringList** &complement) const
- **CStringList** ()
 - Empty CStringList (p. 319) ctor.*
- **CStringList** (const **CStringList** &list)
- **CStringList** (const stringlist &list)
- **CStringList** (const CStringArray &vect)
- **CStringList** (const stringarray &vect)
- virtual void **Dump** (ostream &fOut=cerr) const
 - Dump fonction.*
- virtual void **Erase** (const string &str)
- virtual void **Erase** (CStringList::iterator it)
- virtual bool **Exists** (const string &str) const
- virtual bool **ExistsNoCase** (const string &str) const
- virtual void **ExtractKeys** (const string &str, const string &delim, bool bRemoveAll=true)
- virtual void **ExtractStrings** (const string &str, const char delim, bool bRemoveAll=true)
- virtual void **ExtractStrings** (const string &str, const string &delim, bool bRemoveAll=true)
- virtual int32_t **FindIndex** (const string &str, bool compareNoCase=false) const
- virtual void **Insert** (const **CStringList** &list, bool bEnd=true)
- virtual void **Insert** (const string &str, bool bEnd=true)
- virtual void **Insert** (const CStringArray &vect, bool bEnd=true)
- virtual void **Insert** (const stringarray &vect, bool bEnd=true)
- virtual void **Insert** (const stringlist &lst, bool bEnd=true)
- virtual void **InsertUnique** (const string &str, bool bEnd=true)
- virtual void **InsertUnique** (const **CStringList** &lst, bool bEnd=true)
- virtual void **InsertUnique** (const CStringArray *vect, bool bEnd=true)
- virtual void **InsertUnique** (const CStringArray &vect, bool bEnd=true)
- virtual void **InsertUnique** (const stringarray &vect, bool bEnd=true)
- virtual void **InsertUnique** (const stringlist &lst, bool bEnd=true)
- virtual bool **Intersect** (const **CStringList** &array, **CStringList** &intersect) const
- virtual const **CStringList** & **operator=** (const **CStringList** &lst)
- virtual const **CStringList** & **operator=** (const CStringArray &vect)
- virtual const **CStringList** & **operator=** (const stringarray &vect)

- virtual const **CStringList** & **operator=** (const stringlist &lst)
- virtual void **RemoveAll** ()
- virtual string **Tostring** (const string &delim="," , bool useBracket=true) const
- virtual ~**CStringList** ()

Destructor.

6.112.1 Detailed Description

A list of strings management class.

Version

1.0

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

- List.h
- List.cpp

6.113 brathl::CStringMap Class Reference

```
#include <List.h>
```

Public Member Functions

- **CStringMap** ()
CStringMap (p. 320) ctor.
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CStringMap::iterator it)
- virtual bool **Erase** (const string &key)
- virtual string **Exists** (const string &key) const
- virtual void **GetKeys** (CStringArray &keys, bool bRemoveAll=true) const
- virtual string **Insert** (const string &key, const string &str, bool withExcept=true)
- virtual void **Insert** (const **CStringMap** &strmap, bool withExcept=true)
- virtual string **IsValue** (const string &value)
- virtual void **RemoveAll** ()
- virtual ~**CStringMap** ()
CStringMap (p. 320) dtor.

6.113.1 Detailed Description

a set of string value management classes.

Version

1.0

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

- List.h
- List.cpp

6.114 CTimeChangeEvent Class Reference

```
#include <TimeCtrl.h>
```

Public Member Functions

- virtual wxEvent * **Clone** ()
- **CTimeChangeEvent** ()
- **CTimeChangeEvent** (wxEventType type, wxWindowID id=-1, const wxString &value=wxT(""))
- **CTimeChangeEvent** (const **CTimeChangeEvent** &event)
- wxString **GetValue** () const
- void **SetValue** (const wxString &value)

6.114.1 Detailed Description

This custom time change event triggers whenever the value in the text control changes.

6.114.2 Constructor & Destructor Documentation

6.114.2.1 CTimeChangeEvent::CTimeChangeEvent ()

Default constructor

Referenced by Clone().

6.114.2.2 CTimeChangeEvent::CTimeChangeEvent (wxEventType type, wxWindowID id = -1, const wxString & value = wxT (" "))

Normal constructor

References SetValue().

6.114.2.3 CTimeChangeEvent::CTimeChangeEvent (const CTimeChangeEvent & event)

To cater for **Clone()** (p. 321) function

See Also

Clone() (p. 321)

6.114.3 Member Function Documentation

6.114.3.1 wxEvent * CTimeChangeEvent::Clone () [virtual]

Clone

References CTimeChangeEvent().

6.114.3.2 wxString CTimeChangeEvent::GetValue () const

Get value

6.114.3.3 void CTimeChangeEvent::SetValue (const wxString & value)

Set value

Referenced by CTimeChangeEvent().

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

- TimeCtrl.h
- TimeCtrl.cpp

6.115 CTimeChangeSpinButton Class Reference

```
#include <TimeCtrl.h>
```

Public Member Functions

- **CTimeChangeSpinButton** ()
- **CTimeChangeSpinButton** (CTimeCtrl *timectrl)
- void **OnSpinDown** (wxSpinEvent &event)
- void **OnSpinUp** (wxSpinEvent &event)
- **~CTimeChangeSpinButton** ()

6.115.1 Detailed Description

This control is the spin button of the time picker.

6.115.2 Constructor & Destructor Documentation

6.115.2.1 CTimeChangeSpinButton::CTimeChangeSpinButton ()

Default constructor

6.115.2.2 CTimeChangeSpinButton::CTimeChangeSpinButton (CTimeCtrl * *timectrl*)

Normal constructor

6.115.2.3 CTimeChangeSpinButton::~~CTimeChangeSpinButton ()

Destructor

6.115.3 Member Function Documentation

6.115.3.1 void CTimeChangeSpinButton::OnSpinDown (wxSpinEvent & *event*)

See Also

OnSpinUp(wxSpinEvent& event) (p. 322)

6.115.3.2 void CTimeChangeSpinButton::OnSpinUp (wxSpinEvent & *event*)

These functions are called when the spin button is pressed

Parameters

<i>wxSpinEvent&</i>	
-------------------------	--

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

- TimeCtrl.h
- TimeCtrl.cpp

6.116 brathl::CTools Class Reference

```
#include <Tools.h>
```

Static Public Member Functions

- static double **Abs** (double X)
- static string **AbsolutePath** (const string &partialPath)
- static double **ACos** (double X)
- static double **ACosD** (double X)
- static double **And** (double X, double Y)
- static bool **AreEqual** (double X, double Y)
- static bool **AreEqual** (double X, double Y, double compareEpsilon)
- static bool **AreValidMercatorLatitude** (double lat)
- static string **BaseName** (const string &fileName)
- static string **BeforeFirst** (const string &str, const char ch)
- static double **BitwiseAnd** (double X, double Y)
- static double **BitwiseNot** (double X)
- static double **BitwiseOr** (double X, double Y)
- static bool **CastValue** (int32_t &Dest, const double Source)
- static double **Ceil** (double X)
- static int **Compare** (double X, double Y, double compareEpsilon=CTools::m_CompareEpsilon)
- static bool **Compare** (const char *str1, const char *str2)
- static bool **CompareNoCase** (const char *str1, const char *str2)
- static bool **CompareNoCase** (const string &str1, const string &str2)
- static double **Cos** (double X)
- static double **CosD** (double X)
- static double **Deg2Rad** (double X)
- static void **DeleteObject** (CBratObject *ob)
- static bool **DirectoryExists** (const string &Name)
- static string **DirName** (const string &fileName)
- static double **DistanceKmOnUnitSphere** (double lat1, double long1, double lat2, double long2)
- static double **DistanceOnUnitSphere** (double lat1, double long1, double lat2, double long2)
- static double **Divide** (double X, double Y)
- static void **DoIncrementalStats** (double NewValue, double &Count, double &Mean, double &StdDev, double &Min, double &Max)
- static string **DoubleToStr** (double d, int32_t precision=10)
- static double **Exp** (double X)
- static string **ExpandShellVar** (const string &value)
- static string **ExpandVariables** (const string &valueIn, const map< string, string > *varValues, bool recurse=false, char beginning='%', uint32_t *numberVarsExpanded=NULL, bool withExcept=false, string *errorMsg=NULL)
- static string **ExpandVariables** (const string &valueIn, const map< string, string > *varValues, const map< string, string > *fieldAliases, bool recurse=false, char beginning='%', uint32_t *numberVarsExpanded=NULL, bool withExcept=false, string *errorMsg=NULL)
- static void **ExtractVector** (const double *vectorIn, uint32_t *shape, uint32_t nDims, uint32_t *start, uint32_t *edges, double *vectorOut)
- static bool **FileExists** (const string &Name)
- static string **FileExtension** (const string &fileName)
- static void **FinalizeIncrementalStats** (double Count, double &Mean, double &StdDev, double &Min, double &Max, double DefaultValue=m_defaultValueDOUBLE)
- static void **Find** (const string &inText, const string ®exPattern, vector< string > &stringFound)
- static void **FindAliases** (const string &inText, vector< string > &aliasesFound, bool onlyName=false, const string &beginning="%", bool recurse=false, const map< string, string > *varValues=NULL, const map< string, string > *fieldAliases=NULL, bool withExcept=false, string *errorMsg=NULL)

- static string **FindDataFile** (const string &Name)
- static string **FindFileInPath** (const string &filename, const string &path)
- static int32_t **FindNoCase** (const string &src, const string &findWhat, uint32_t pos=0)
- static int32_t **FindNoCase** (const char *src, const char *findWhat, uint32_t pos=0)
- static void **FindWord** (const string &inText, vector< string > &wordsFound)
- static string **FloatToStr** (float f, int32_t precision=10)
- static double **Floor** (double X)
- static int32_t static string **Format** (size_t size, const char *format,...) __attribute__((format(printf
- static int32_t static string
static string **Format** (const char *format,...) __attribute__((format(printf
- static int32_t static string
static string static string **Format** (size_t size, const char *format, va_list args)
- static double **Frac** (double value)
- static string **GetDataDir** ()
- static uint32_t **GetProductValues** (uint32_t *shape, uint32_t nbDims)
- static double **Int** (double dValue)
- static string **IntToStr** (int32_t i)
- static double **IsBounded** (double **Min**, double X, double **Max**)
- static double **IsBoundedStrict** (double **Min**, double X, double **Max**)
- static double **IsDefaultFloat** (double X)
- static bool **IsDefaultValue** (const float value)
- static bool **IsDefaultValue** (const double value)
- static bool **IsDefaultValue** (const int8_t value)
- static bool **IsDefaultValue** (const uint8_t value)
- static bool **IsDefaultValue** (const int16_t value)
- static bool **IsDefaultValue** (const uint16_t value)
- static bool **IsDefaultValue** (const int32_t value)
- static bool **IsDefaultValue** (const uint32_t value)
- static bool **IsDefaultValue** (const int64_t value)
- static bool **IsDefaultValue** (const uint64_t value)
- static bool **IsEmpty** (const char *pstrString)
- static bool **IsEven** (uint32_t value)
- static bool **IsEven** (int32_t value)
- static int **IsInf** (double X)
- static bool **IsLongitudeCircular** (double min, double max, double compareEpsilon=CTools::m_CompareEpsilon)
- static int **IsNan** (double X)
- static bool **IsOdd** (uint32_t value)
- static bool **IsOdd** (int32_t value)
- static bool **IsZero** (double X)
- static bool **LoadAndCheckUdUnitsSystem** (string &errorMsg)
- static double **Log** (double X)
- static double **Log10** (double X)
- static string **LongToStr** (int64_t i)
- static string **MakeCorrectPath** (const string &path)
- static double **Max** (double X1, double X2)
- static double **Min** (double X1, double X2)
- static double **Minus** (double X, double Y)
- static double **Mod** (double X, double Y)
- static double **Multiply** (double X, double Y)
- static double **NormalizeLongitude** (double **Floor**, double Longitude)
- static double **Or** (double X, double Y)
- static double **Plus** (double X, double Y)
- static double **Pow** (double X, double Y)
- static double **Rad2Deg** (double X)

- static char * **RemoveAllSpaces** (char *str)
- static string **RemoveCharSurroundingNumber** (const string &str, const char c1= '(', const char c2= ')')
- static string **Replace** (const string &inText, const string ®exPattern, const string replaceString)
- static void **ReplaceAliases** (const string &in, string &out, vector< string > *aliases=NULL)
- static void **ReplaceAliases** (const string &in, const string &replacedBy, string &out, vector< string > *aliases=NULL)
- static string **ReplaceString** (const string &inText, const vector< string > &findString, const vector< string > &replaceWords)
- static string **ReplaceWord** (const string &inText, const vector< string > &findWords, const vector< string > &replaceWords)
- static string **ReplaceWord** (const string &inText, const string &findWords, const string &replaceWords)
- static int32_t **RFindNoCase** (const string &src, const string &findWhat, uint32_t pos=0)
- static int32_t **RFindNoCase** (const char *src, const char *findWhat, uint32_t pos=0)
- static double **Rnd** (double value, double precision)
- static double **Round** (double value)
- static void **SetDataDir** (const string &DataDir)
- static void **SetDataDirForExecutable** (const char *argv0)
- static void **SetDefaultValue** (float &value)
- static void **SetDefaultValue** (double &value)
- static void **SetDefaultValue** (int8_t &value)
- static void **SetDefaultValue** (uint8_t &value)
- static void **SetDefaultValue** (int16_t &value)
- static void **SetDefaultValue** (uint16_t &value)
- static void **SetDefaultValue** (int32_t &value)
- static void **SetDefaultValue** (uint32_t &value)
- static void **SetDefaultValue** (int64_t &value)
- static void **SetDefaultValue** (uint64_t &value)
- static double **Sign** (double X)
- static double **Sin** (double X)
- static double **Sinc** (double x)
- static double **SinD** (double X)
- static string **SlashesDecode** (const string &str, const string &exclude="", bool decodeliterals=true)
- static string **SlashesEncode** (const string &str, const string &exclude="", const string &literals="", bool hexadecimal=true)
- static int32_t **snprintf** (char *str, size_t size, const char *format,...) __attribute__((format(printf
- static double **Sqr** (double X)
- static double **Sqrt** (double X)
- static int32_t **StrCaseCmp** (const char *str1, const char *str2)
- static bool **StringCompare** (const string &s1, const string &s2)
- static string **StringRemoveAllSpaces** (const string &str)
- static string **StringReplace** (const string &str, char c, char replaceBy)
- static string **StringReplace** (const string &str, const string &c, const string &replaceBy, bool compareNoCase=false)
- static void **StringToAlias** (const string &in, string &out, const char beginning)
- static string **StringToLower** (const string &str)
- static string **StringToUpper** (const string &str)
- static string **StringTrim** (const string &str)
- static double **StrToDouble** (const string &value)
- static float **StrToFloat** (const string &value)
- static int32_t **StrToInt** (const string &s)
- static int64_t **StrToInt64** (const string &s)
- static int64_t **StrToLong** (const string &s)
- static uint64_t **StrToUInt64** (const string &s)
- static void **SwapValue** (int32_t &value)
- static void **SwapValue** (int16_t &value)

- static void **SwapValue** (float &value)
- static void **SwapValue** (double &value)
- static double **Tan** (double X)
- static double **TanD** (double X)
- static char * **ToLower** (char *str)
- static char **ToLower** (const char chr)
- static string **ToString** (const char *s, size_t len=string::npos)
- static char * **ToUpper** (char *str)
- static char **ToUpper** (const char chr)
- static string **TrailingZeroesTrim** (const string &Text, bool dotTrim=true)
- static char * **Trim** (char *str)
- static double **UnaryMinus** (double X)
- static double **UnaryNot** (double X)
- static double **UnconvertLat** (const string &value)
- static double **UnconvertLon** (const string &value, bool normalize=true)
- static int32_t **VectorContiguousBlock** (uint32_t ndims, const uint32_t *const shape, const uint32_t *const edges, uint32_t *const countContinuousBlock)
- static uint32_t **VectorOffset** (uint32_t *shape, uint32_t ndims, const uint32_t *coord)
- static bool **Xor** (bool p, bool q)

Static Public Attributes

- static const double **m_CompareEpsilon** = 1.0E-70
- static const char **m_defaultValueCHAR** = '\0'
default values for chars
- static const double **m_defaultValueDOUBLE** = 18446744073709551616.0
default values for double
- static const float **m_defaultValueFLOAT** = 18446744073709551616.0F
default values for float
- static const int16_t **m_defaultValueINT16** = 0x7FFF
default values for int 16 bits
- static const int32_t **m_defaultValueINT32** = 0xFFFFFFFF
default values for int 32 bits
- static const int64_t **m_defaultValueINT64** = 0xFFFFFFFFFFFFFFFFLL
default values for unsigned int 64 bits
- static const int8_t **m_defaultValueINT8** = 0x7F
default values for int 8 bits
- static const char * **m_defaultValueString** = ""
default values for string
- static const uint16_t **m_defaultValueUINT16** = 0xFFFFU
default values for unsigned int 16 bits
- static const uint32_t **m_defaultValueUINT32** = 0xFFFFFFFFU
default values for unsigned int 32 bits
- static const uint64_t **m_defaultValueUINT64** = 0xFFFFFFFFFFFFFFFFULL
default values for unsigned int 64 bits
- static const uint8_t **m_defaultValueUINT8** = 0xFFU
default values for unsigned int 8 bits
- static const double **m_deltaLatitudeMercator** = 1.0E-7

6.116.1 Detailed Description

Tools management class.

This class provides various static utility methods

Version

1.0

6.116.2 Member Function Documentation

6.116.2.1 double brathl::CTools::Abs (double *X*) [static]

Find the absolute value of a number. Takes default values into account

Parameters

in	<i>X</i>	: Number involved
----	----------	-------------------

Returns

Result of operation

6.116.2.2 string brathl::CTools::AbsolutePath (const string & *partialPath*) [static]

Creates an absolute or full path name for the specified relative path name.

- change path separator in a suitable path separator ('\ or '/' depending on the system)
- skip trailing "../", if any
- remove back references: translate dir1/./dir2 to dir2

Parameters

in	<i>partialPath</i>	: the relative path
----	--------------------	---------------------

Returns

the absolute path name, or empty string if there is an error (for example, if the value passed in relPath includes a drive letter that is not valid or cannot be found, or if the length of the created absolute path name is greater than the BRATHL_PATH_MAX defined in **brathl.h** (p. 369))

6.116.2.3 double brathl::CTools::ACos (double *X*) [static]

Do the arc cosine of a number expressed in radians. Takes default values into account

Parameters

in	<i>X</i>	: Number involved
----	----------	-------------------

Returns

Result of operation

Referenced by ACosD().

6.116.2.4 double brathl::CTools::ACosD (double *X*) [static]

Do the arc cosine of a number expressed in degrees. Takes default values into account

Parameters

<i>in</i>	<i>X</i>	: Number involved
-----------	----------	-------------------

Returns

Result of operation

References ACos().

6.116.2.5 double brathl::CTools::And (double *X*, double *Y*) [static]

Do a logical and on two numbers. Takes default values into account

Parameters

<i>in</i>	<i>X</i>	: Number involved
<i>in</i>	<i>Y</i>	: Number involved

Returns

Result of operation

6.116.2.6 string brathl::CTools::BaseName (const string & *fileName*) [static]

Gets a base file name from a string

Parameters

<i>in</i>	<i>path</i>	: full path
-----------	-------------	-------------

Returns

the base file name (no extension), or empty string, or : '.' returns '.', './' returns '.', '/' returns '/', '..' returns '..', '../' returns '..' 'abc/def/' returns 'def'

6.116.2.7 double brathl::CTools::BitwiseAnd (double *X*, double *Y*) [static]

Do a bitwise AND operation an integer. The numbers are taken as signed integers (int32_t). Then a bitwise AND is computed and the integer is converted back to a float. If the parameters are default values or do not fall in integer range, a default value is returned.

Parameters

<i>in</i>	<i>X</i>	: Number involved
<i>in</i>	<i>Y</i>	: Number involved

Returns

Result of operation

6.116.2.8 double brathl::CTools::BitwiseNot (double *X*) [static]

Complement an integer. The number is taken as a signed integer (int32_t). Then a bitwise not is computed and the integer is converted back to a float. If the parameter is a default values or do not fall in integer range, a default value is returned.

Parameters

in	X	: Number involved
----	---	-------------------

Returns

Complemented number

6.116.2.9 double brathl::CTools::BitwiseOr (double X, double Y) [static]

Do a bitwise OR operation an integer. The numbers are taken as signed integers (int32_t). Then a bitwise OR is computed and the integer is converted back to a float. If the parameters are default values or do not fall in integer range, a default value is returned.

Parameters

in	X	: Number involved
in	Y	: Number involved

Returns

Result of operation

6.116.2.10 double brathl::CTools::Ceil (double X) [static]

Find the integral value part over of a number. Takes default values into account

Parameters

in	X	: Number involved
----	---	-------------------

Returns

Result of operation

6.116.2.11 double brathl::CTools::Cos (double X) [static]

Do the cosine of a number expressed in radians. Takes default values into account

Parameters

in	X	: Number involved
----	---	-------------------

Returns

Result of operation

6.116.2.12 double brathl::CTools::CosD (double X) [static]

Do the cosine of a number expressed in degrees. Takes default values into account

Parameters

in	X	: Number involved
----	---	-------------------

Returns

Result of operation

6.116.2.13 `double bratl::CTools::Deg2Rad (double X)` `[static]`

Convert degrees to radians. Takes default values into account

Parameters

<code>in</code>	<code><i>X</i></code>	: Number involved
-----------------	-----------------------	-------------------

Returns

Result of operation

Referenced by TanD().

6.116.2.14 `bool bratl::CTools::DirectoryExists (const string & Name)` `[static]`

Indicates if a directory exists

Parameters

<code>in</code>	<code><i>Name</i></code>	: Directory name
-----------------	--------------------------	------------------

Returns

Returns true if directory exists

6.116.2.15 `string bratl::CTools::DirName (const string & fileName)` `[static]`

Gets a directory name from a string

Parameters

<code>in</code>	<code><i>path</i></code>	: full path
-----------------	--------------------------	-------------

Returns

the directory name, or '.' if path has only one component

6.116.2.16 `double bratl::CTools::Divide (double X, double Y)` `[static]`

Divide two numbers. Takes default values into account

Parameters

<code>in</code>	<code><i>X</i></code>	: Number involved
<code>in</code>	<code><i>Y</i></code>	: Number involved

Returns

Result of operation

6.116.2.17 `void brathl::CTools::DoIncrementalStats (double NewValue, double & Count, double & Mean, double & StdDev, double & Min, double & Max) [static]`

Do incremental statistics. Incremental statistics are done to avoid memory consumption needed when we do 'classical' stats: an array of all the values involved with statistics must be kept before computing them. After first call to this the parameters must not be modified until end of statistics or result will be unpredictable.

Parameters

<i>in</i>	<i>NewValue</i>	: New value to take into account for statistics. Only valid values are kept; valid values are those different from default value (#IsDefaultValue#)
	<i>in/out</i>	Count : number of valid data used for stats. Valid data is a number which is not a default value. On first call, this parameter must be 0 or a default value. And it is not modified since the first valid value.
	<i>in/out</i>	Mean : Incremental mean
	<i>in/out</i>	StdDev : Temporary value used to compute standard deviation
	<i>in/out</i>	Min : Minimum value
	<i>in/out</i>	Max : Maximum value

6.116.2.18 `string brathl::CTools::DoubleToStr (double d, int32_t precision = 10) [static]`

Convert an double to string

Parameters

<i>in</i>	<i>value</i>	: double to be converted
-----------	--------------	--------------------------

Returns

coanverted value or empty string if no possible conversion.

6.116.2.19 `double brathl::CTools::Exp (double X) [static]`

Find exponential of a number. Takes default values into account

Parameters

<i>in</i>	<i>X</i>	: Number involved
-----------	----------	-------------------

Returns

Result of operation

References IsInf().

6.116.2.20 `string brathl::CTools::ExpandShellVar (const string & value) [static]`

Expands shell variables (i.e. \${HOME}). If the '\$' character is preceded by '\', it's taken into account as a common character and not as a shell variable identifier. Shell variables beginning by '+' are expanded in uppercase. Shell variables beginning by '-' are expanded in lowercase.

Parameters

<i>in</i>	<i>value</i>	: The string to expand
-----------	--------------	------------------------

Returns

the newly expanded string.

References ExpandVariables().

Referenced by bratl::CParameter::AddValue().

6.116.2.21 `string bratl::CTools::ExpandVariables (const string & valueIn, const map< string, string > * varValues, bool recurse = false, char beginning = ' % ', uint32_t * numberVarsExpanded = NULL, bool withExcept = false, string * errorMsg = NULL) [static]`

Expand variables (i.e. %{VAR}). If the " character is preceded by '\', it's taken into account as a common character and not as a variable identifier. Variables beginning by '+' are expanded in uppercase. Variables beginning by '-' are expanded in lowercase.

Parameters

in	<i>value</i>	: The string to expand
in	<i>VarValues</i>	: The values of the variables. If NULL, the environment variables are taken.
in	<i>Begining</i>	: Char identifying the beginning of a var reference
in	<i>Recurse</i>	: If true, variable expanded can contain references to other variables which are then expanded.

Returns

the newly expanded string.

Referenced by ExpandShellVar(), and bratl::CParameter::SetAliases().

6.116.2.22 `bool bratl::CTools::FileExists (const string & Name) [static]`

Indicates if a file exists

Parameters

in	<i>Name</i>	: File name
----	-------------	-------------

Returns

Returns true if file exists and is readable

6.116.2.23 `string bratl::CTools::FileExtension (const string & fileName) [static]`

Gets a file name extension.

Parameters

in	<i>filename</i>	: file name
----	-----------------	-------------

Returns

the extension, or empty string if none

6.116.2.24 `void bratl::CTools::FinalizeIncrementalStats (double Count, double & Mean, double & StdDev, double & Min, double & Max, double DefaultValue = m_defaultValueDOUBLE) [static]`

Terminates incremental statistics. Computes the final value of standard deviation

Parameters

in	<i>Count</i>	: number of valid data used for stats. If count is 0 or default value, all other output parameters are set to default value.
	<i>in/out</i>	Mean : Computed mean or default value (see Count)
	<i>in/out</i>	StdDev : On output, actual value of standard deviation
	<i>in/out</i>	Min : Computed min or default value (see Count)
	<i>in/out</i>	Max : Computed max or default value (see Count)
in	<i>DefaultValue</i>	: Default value wanted Value to put in output parameters if no stats can be done (no valid data: count is 0 or default value m_defaultValueDOUBLE (p. 326)#).

6.116.2.25 string brathl::CTools::FindDataFile (const string & *Name*) [static]

Finds a file path known only by its name. The path is retrieved from compilation (intallation prefix) or by environment variable.

Parameters

in	<i>Name</i>	: File name
----	-------------	-------------

Returns

Returns the path of found file or an empty string if not found

Referenced by brathl::CMission::LoadAliasName().

6.116.2.26 string brathl::CTools::FindFileInPath (const string & *filename*, const string & *path*) [static]

Finds a file location known only by its name using the give path. The path should be similar to what can be used for the PATH environment variable on the current system.

Parameters

in	<i>filename</i>	: File name
in	<i>path</i>	: Search path

Returns

Returns the full path to the file or an empty string if not found

6.116.2.27 double brathl::CTools::Floor (double *X*) [static]

Find the integral value part below of a number. Takes default values into account

Parameters

in	<i>X</i>	: Number involved
----	----------	-------------------

Returns

Result of operation

6.116.2.28 string brathl::CTools::Format (size_t *size*, const char * *format*, ...) [static]

Write formatted data to a string. WARNING : this method use vsnprintf if vsnprintf is defined, otherwise vsprintf is used and 'size' parameter is ignored

Parameters

in	<i>size</i>	: maximum number of characters to store
in	<i>format</i>	: format-control string
in	...	: optional arguments

Returns

formatted string

Referenced by brathl::CDate::AsString(), brathl::BuildExistingInternalFileKind(), brathl::CFileParams::CheckCount(), brathl::CDate::CvDate(), brathl::CFloatArray::Dump(), brathl::CDoubleArray::Dump(), brathl::CDoubleMap::Dump(), brathl::CObDoubleMap::Dump(), brathl::CDoublePtrDoubleMap::Dump(), brathl::CDataSet::EraseFieldSet(), brathl::CBratAlgorithmGeosVelGrid::GetInputParamDesc(), brathl::CBratAlgorithmGeosVelAtp::GetInputParamDesc(), brathl::CBratAlgoFilterMedian1D::GetInputParamDesc(), brathl::CBratAlgoFilterLoess1D::GetInputParamDesc(), brathl::CBratAlgoFilterLoess2D::GetInputParamDesc(), brathl::CBratAlgoFilterMedian2D::GetInputParamDesc(), brathl::CBratAlgorithmGeosVelGrid::GetInputParamFormat(), brathl::CBratAlgorithmGeosVelAtp::GetInputParamFormat(), brathl::CBratAlgoFilterMedian1D::GetInputParamFormat(), brathl::CBratAlgoFilterLoess2D::GetInputParamFormat(), brathl::CBratAlgoFilterLoess1D::GetInputParamFormat(), brathl::CBratAlgoFilterMedian2D::GetInputParamFormat(), brathl::CBratAlgorithmGeosVelGrid::GetInputParamUnit(), brathl::CBratAlgorithmGeosVelAtp::GetInputParamUnit(), brathl::CBratAlgoFilterMedian1D::GetInputParamUnit(), brathl::CBratAlgoFilterMedian2D::GetInputParamUnit(), brathl::CBratAlgoFilterLoess2D::GetInputParamUnit(), brathl::CBratAlgoFilterLoess1D::GetInputParamUnit(), brathl::CParameter::GetValue(), brathl::CUIntMap::Insert(), brathl::CDataSet::InsertFieldSet(), brathl::CProductErsWAP::IsHighResolutionField(), brathl::CFile::Open(), brathl::CFile::ReadToBuffer(), brathl::CBratAlgoFilterLanczos1D::Run(), brathl::CBratAlgoFilterGaussian1D::Run(), brathl::CBratAlgoFilterMedian1D::Run(), brathl::CBratAlgoFilterLoess1D::Run(), brathl::CDatePeriod::SetFrom(), brathl::CDatePeriod::SetTo(), SlashesDecode(), SlashesEncode(), brathl::CFile::WriteChar(), brathl::CFile::WriteFromBuffer(), and brathl::CFile::WriteString().

6.116.2.29 string brathl::CTools::Format (const char * *format*, ...) [static]

Write formatted data to a string. WARNING : this method use vsnprintf if vsnprintf is defined, otherwise vsprintf is used and 'size' parameter is ignored

Parameters

in	<i>format</i>	: format-control string
in	...	: optional arguments

Returns

formatted string

6.116.2.30 string brathl::CTools::Format (size_t *size*, const char * *format*, va_list *args*) [static]

Write formatted data to a string. WARNING : this method use vsnprintf if vsnprintf is defined, otherwise vsprintf is used and 'size' parameter is ignored

Parameters

in	<i>size</i>	: maximum number of characters to store
in	<i>format</i>	: format-control string
in	<i>args</i>	: optional arguments

Returns

formatted string

6.116.2.31 `string brathl::CTools::GetDataDir () [static]`

Returns the constant data directory defined at compilation time, by environment variable, or set by application.

Returns

Returns the path of found file or an empty string if not found

Referenced by `brathl::CMission::LoadAliasName()`.

6.116.2.32 `string brathl::CTools::IntToStr (int32_t i) [static]`

Convert an int to string

Parameters

<i>in</i>	<i>value</i>	: int to be converted
-----------	--------------	-----------------------

Returns

converted value or empty string if no possible conversion.

6.116.2.33 `double brathl::CTools::IsBounded (double Min, double X, double Max) [static]`

Indicates if a number is comprised between two others. Takes default values into account

Parameters

<i>in</i>	<i>Min</i>	: Lower bound
<i>in</i>	<i>X</i>	: Number involved
<i>in</i>	<i>Max</i>	: Upper bound

Returns

Result of operation: 0 if not $Min \leq X \leq Max$.

6.116.2.34 `double brathl::CTools::IsBoundedStrict (double Min, double X, double Max) [static]`

Indicates if a number is comprised between two others. Takes default values into account

Parameters

<i>in</i>	<i>Min</i>	: Lower bound
<i>in</i>	<i>X</i>	: Number involved
<i>in</i>	<i>Max</i>	: Upper bound

Returns

Result of operation: 0 if not $Min < X < Max$.

6.116.2.35 `double brathl::CTools::IsDefaultFloat (double X) [static]`

Checks a default value.

Parameters

<i>in</i>	<i>X</i>	: Number involved
-----------	----------	-------------------

Returns

0.0 if X is not a default value, 1.0 otherwise

6.116.2.36 `int32_t brathl::CTools::IsInf (double X) [static]`

Indicates if a number is infinite.

Parameters

<code>in</code>	<code>X</code>	: Number involved
-----------------	----------------	-------------------

Returns

0 if X in finite 1 if infinite

Referenced by `Exp()`, `Pow()`, `Sqr()`, and `Tan()`.

6.116.2.37 `int32_t brathl::CTools::IsNaN (double X) [static]`

Indicates if a value is a valid number.

Parameters

<code>in</code>	<code>X</code>	: Number involved
-----------------	----------------	-------------------

Returns

0 if X is valid, 1 if X is not a number

Referenced by `Tan()`.

6.116.2.38 `double brathl::CTools::Log (double X) [static]`

Find the natural logarithm of a number. Takes default values into account

Parameters

<code>in</code>	<code>X</code>	: Number involved
-----------------	----------------	-------------------

Returns

Result of operation

6.116.2.39 `double brathl::CTools::Log10 (double X) [static]`

Find the decimal logarithm of a number. Takes default values into account

Parameters

<code>in</code>	<code>X</code>	: Number involved
-----------------	----------------	-------------------

Returns

Result of operation

6.116.2.40 `string brathl::CTools::MakeCorrectPath (const string & path) [static]`

Cleans a path variable

- change path separator in a suitable path separator ('\' or '/' depending on the system)
- skip trailing "...", if any
- remove back references: translate dir1/./dir2 to dir2

Parameters

<i>in</i>	<i>path</i>	: The string to clean
-----------	-------------	-----------------------

Returns

the newly cleaned string.

6.116.2.41 double brathl::CTools::Max (double *X1*, double *X2*) [static]

Find the maximum value of two numbers. Takes default values into account

Parameters

<i>in</i>	<i>X1</i>	: Number involved
<i>in</i>	<i>X2</i>	: Number involved

Returns

Result of operation

Referenced by brathl::CCriteriaLatLon::GetMinOrMaxLon().

6.116.2.42 double brathl::CTools::Min (double *X1*, double *X2*) [static]

Find the minimum value of two numbers. Takes default values into account

Parameters

<i>in</i>	<i>X1</i>	: Number involved
<i>in</i>	<i>X2</i>	: Number involved

Returns

Result of operation

Referenced by brathl::CCriteriaLatLon::GetMinOrMaxLon().

6.116.2.43 double brathl::CTools::Minus (double *X*, double *Y*) [static]

Subtracts one number from another. TAKES default values into account

Parameters

<i>in</i>	<i>X</i>	: Number involved
<i>in</i>	<i>Y</i>	: Number involved

Returns

Result of operation

6.116.2.44 double brathl::CTools::Mod (double *X*, double *Y*) [static]

Find the modulus of a number divided by another. Takes default values into account

Parameters

in	X	: Number involved
in	Y	: Divider

Returns

Result of operation

6.116.2.45 `double brathl::CTools::Multiply (double X, double Y) [static]`

Multiply two numbers. Takes default values into account

Parameters

in	X	: Number involved
in	Y	: Number involved

Returns

Result of operation

6.116.2.46 `double brathl::CTools::NormalizeLongitude (double Floor, double Longitude) [static]`

Find a number satisfying the condition $\text{Floor} \leq \text{Longitude} < \text{Floor} + 360$. Takes default values into account

Parameters

in	<i>Floor</i>	: Base longitude
in	<i>Longitude</i>	: Longitude to normalize

Returns

Result of operation

6.116.2.47 `double brathl::CTools::Or (double X, double Y) [static]`

Do a logical or on two numbers. Takes default values into account

Parameters

in	X	: Number involved
in	Y	: Number involved

Returns

Result of operation

6.116.2.48 `double brathl::CTools::Plus (double X, double Y) [static]`

Add two numbers. Takes default values into account

Parameters

in	X	: Number involved
in	Y	: Number involved

Returns

Result of operation

6.116.2.49 `double brathl::CTools::Pow (double X, double Y) [static]`

Find the power of a number by another. Takes default values into account

Parameters

<i>in</i>	<i>X</i>	: Number involved
<i>in</i>	<i>Y</i>	: Power. Can be a integral or decimal

Returns

Result of operation

References `IsInf()`.

6.116.2.50 `double brathl::CTools::Rad2Deg (double X) [static]`

Convert radians to degrees. Takes default values into account

Parameters

<i>in</i>	<i>X</i>	: Number involved
-----------	----------	-------------------

Returns

Result of operation

6.116.2.51 `char * brathl::CTools::RemoveAllSpaces (char * str) [static]`

Remove all the blank characters in a string. Blank characters are identified by the function `isspace (3C)`.

Parameters

<i>str</i>	[in/out] : string to be modified
------------	----------------------------------

Returns

a pointer to the string

Referenced by `StringRemoveAllSpaces()`.

6.116.2.52 `string brathl::CTools::RemoveCharSurroundingNumber (const string & str, const char c1 = ' (', const char c2 = ') ') [static]`

Removes characters *c1* and *c2*, if these characters surround an number (integer or decimal). For example: `RemoveCharSurroundingNumber("ABCD (125)", '(', ')')` will return "ABCD 125" `RemoveCharSurroundingNumber("ABCD (+125.63)", '(', ')')` will return "ABCD +125.63" `RemoveCharSurroundingNumber("ABCD (-45) (XYZ*2)", '(', ')')` will return "ABCD -45 (XYZ*2)" `RemoveCharSurroundingNumber("ABCD ((-45)))", '(', ')')` will return "(ABCD (-45))"

Parameters

<i>in</i>	<i>str</i>	: The string to modify
<i>in</i>	<i>c1</i>	: the first surrounding char
<i>in</i>	<i>c2</i>	: the last surrounding char

Returns

the newly modified string.

6.116.2.53 void brathl::CTools::SetDataDir (const string & *DataDir*) [static]

Explicitly set the Data Directory.

Parameters

in	<i>DataDir</i>	: Full path to data directory.
----	----------------	--------------------------------

6.116.2.54 void brathl::CTools::SetDataDirForExecutable (const char * *argv0*) [static]

Explicitly set the Data Directory based on a relative path to the current executable. The Data Directory will be set to './data' relative to the location of the executable.

Parameters

in	<i>argv0</i>	: pass argv[0] that you got from main(int argc, char *argv[]).
----	--------------	--

6.116.2.55 double brathl::CTools::Sign (double *X*) [static]

Find the sign of a number (1 if positive or null, -1 if negative). Takes default values into account

Parameters

in	<i>X</i>	: Number involved
----	----------	-------------------

Returns

Result of operation

6.116.2.56 double brathl::CTools::Sin (double *X*) [static]

Do the sine of a number expressed in radians. Takes default values into account

Parameters

in	<i>X</i>	: Number involved
----	----------	-------------------

Returns

Result of operation

6.116.2.57 double brathl::CTools::SinD (double *X*) [static]

Do the sine of a number expressed in degrees. Takes default values into account

Parameters

in	<i>X</i>	: Number involved
----	----------	-------------------

Returns

Result of operation

6.116.2.58 `string brathl::CTools::SlashesDecode (const string & str, const string & exclude = " ", bool decodeliterals = true) [static]`

Takes a string with escaped charters including decimal and hexadecimal escapes and decodes them to the literal charter. This function supports only standard C/C++ escaped literals.

Parameters

in	<i>str</i>	: The string to decode.
in	<i>exclude</i>	: A list of charters to exclude from decoding.
in	<i>decodeliterals</i>	: Set if non standard escaped literals are to be decoded.

Returns

the newly encoded string.

References `Format()`.

6.116.2.59 `string brathl::CTools::SlashesEncode (const string & str, const string & exclude = " ", const string & literals = " ", bool hexadecimal = true) [static]`

This encodes characters that are not printable or can be encode with one of the C/C++ standard escape sequences. The 'exclude' list is a list of chars to exclude from the encoding process. Since the '\0' is used to determine the end of the string and will not be encoded.

Parameters

in	<i>str</i>	: The string to encode.
in	<i>exclude</i>	: A list of charters to exclude from encoding.
in	<i>literals</i>	:A list of printable characters to be included in the encodeing.
	<i>hexadecimal</i>	If true, non-standard, non-printable charecters will be encoded in hexadecimal. If false they will be encoded in octal format.

Returns

the newly encoded string.

References `Format()`.

6.116.2.60 `int32_t brathl::CTools::snprintf (char * str, size_t size, const char * format, ...) [static]`

Write formatted data to a string. WARNING : this method use `vsnprintf` if `vsnprintf` is defined, otherwise `vsprintf` is used and 'size' parameter is ignored

Parameters

out	<i>str</i>	: storage location for output.
in	<i>size</i>	: maximum number of characters to store
in	<i>format</i>	: format-control string
in	...	: optional arguments

Returns

return value of the `vsnprintf` or `vsprintf` - see documentation of these functions

6.116.2.61 `double brathl::CTools::Sqr (double X) [static]`

Find the square value of a number. Takes default values into account

Parameters

<i>in</i>	<i>X</i>	: Number involved
-----------	----------	-------------------

Returns

Result of operation

References IsInf().

6.116.2.62 `double brathl::CTools::Sqrt (double X) [static]`

Find the square root value of a number. Takes default values into account

Parameters

<i>in</i>	<i>X</i>	: Number involved
-----------	----------	-------------------

Returns

Result of operation

6.116.2.63 `int32_t brathl::CTools::StrCaseCmp (const char * str1, const char * str2) [static]`

Compare the two strings *str1* and *str2*, while being unaware of the differences between upper-case and lower-case. This method is thus identical to the function `strcasecmp` (3C) with the following difference : *str1*, *str2* can be NULL, in this case, the string concerned is regarded as a null string.

Parameters

<i>in</i>	<i>str1</i>	: string 1
<i>in</i>	<i>str2</i>	: string 2

Returns

: negative, null (= 0) or positive value if the *str1* is respectively lower, equal or higher than *str2*.

Referenced by `brathl::CParameter::GetValue()`.

6.116.2.64 `string brathl::CTools::StringRemoveAllSpaces (const string & str) [static]`

Remove all the blank characters in a string. Blank characters are identified by the function `isspace` (3C).

Parameters

<i>in</i>	<i>str</i>	: string to be modified
-----------	------------	-------------------------

Returns

the modified string

References `RemoveAllSpaces()`.

6.116.2.65 `string brathl::CTools::StringReplace (const string & str, char c, char replaceBy) [static]`

Replace all tokens of char *c* by char *replaceBy* in a string.

Parameters

in	<i>str</i>	: string to be modified
in	<i>c</i>	: char to replace
in	<i>replaceBy</i>	: char replaced

Returns

the modified string

6.116.2.66 `string bratl::CTools::StringReplace (const string & str, const string & c, const string & replaceBy, bool compareNoCase = false) [static]`

Replace all tokens of string *c* by string *replaceBy* in a string.

Parameters

in	<i>str</i>	: string to be modified
in	<i>c</i>	: string to replace
in	<i>replaceBy</i>	: string replaced

Returns

the modified string

6.116.2.67 `string bratl::CTools::StringToLower (const string & str) [static]`

Set a string object in lowercase

Parameters

<i>str</i>	[in/out] : string to be modified
------------	----------------------------------

Returns

a new string object in lowercase

References ToLower().

Referenced by bratl::CProductEnvisat::IsHighResolutionField().

6.116.2.68 `string bratl::CTools::StringToUpper (const string & str) [static]`

Set a string object in uppercase

Parameters

in	<i>str</i>	: character
----	------------	-------------

Returns

a new string object in uppercase

References ToUpper().

6.116.2.69 `string bratl::CTools::StringTrim (const string & str) [static]`

Remove all the blank characters at the beginning and the end of a string. Blank characters are identified by the function isspace (3C).

Parameters

<i>str</i>	[in/out] : string to be modified
------------	----------------------------------

Returns

a trimmed string

Referenced by brathl::CMission::LoadAliasName(), StrToDouble(), Trim(), UnconvertLat(), and UnconvertLon().

6.116.2.70 `double brathl::CTools::StrToDouble (const string & value) [static]`

Convert an string to double

Parameters

<i>in</i>	<i>value</i>	: string to be converted
-----------	--------------	--------------------------

Returns

coconverted value or CTool::m_defaultValueDOUBLE if no possible conversion.

References StringTrim().

Referenced by UnconvertLat(), and UnconvertLon().

6.116.2.71 `int32_t brathl::CTools::StrToInt (const string & s) [static]`

Convert an string to int

Parameters

<i>in</i>	<i>value</i>	: string to be converted
-----------	--------------	--------------------------

Returns

coconverted value or CTool::m_defaultValueINT if no possible conversion.

Referenced by brathl::CCriteriaCycle::Set(), brathl::CCriteriaPassInt::Set(), brathl::CCriteriaCycle::SetFrom(), brathl::CCriteriaPassInt::SetFrom(), brathl::CCriteriaCycle::SetTo(), and brathl::CCriteriaPassInt::SetTo().

6.116.2.72 `int64_t brathl::CTools::StrToInt64 (const string & s) [static]`

Convert an string to int64

Parameters

<i>in</i>	<i>value</i>	: string to be converted
-----------	--------------	--------------------------

Returns

coconverted value or CTool::m_defaultValueINT if no possible conversion.

6.116.2.73 `uint64_t brathl::CTools::StrToUInt64 (const string & s) [static]`

Convert an string to uint64

Parameters

<i>in</i>	<i>value</i>	: string to be converted
-----------	--------------	--------------------------

Returns

coanverted value or CTool::m_defaultValueINT if no possible conversion.

6.116.2.74 double brathl::CTools::Tan (double *X*) [static]

Do the tangent of a number expressed in radians. Takes default values into account

Parameters

<i>in</i>	<i>X</i> : Number involved
-----------	----------------------------

Returns

Result of operation

References IsInf(), and IsNan().

Referenced by TanD().

6.116.2.75 double brathl::CTools::TanD (double *X*) [static]

Do the tangent of a number expressed in degrees. Takes default values into account

Parameters

<i>in</i>	<i>X</i> : Number involved
-----------	----------------------------

Returns

Result of operation

References Deg2Rad(), and Tan().

6.116.2.76 char * brathl::CTools::ToLower (char * *str*) [static]

Set a string in lowercase

Parameters

<i>str</i> [in/out] : string to be modified

Returns

a pointer to the string

Referenced by StringToLower().

6.116.2.77 char brathl::CTools::ToLower (const char *chr*) [static]

Set a string in lowercase

Parameters

<i>in</i>	<i>chr</i> : character
-----------	------------------------

Returns

the lowercase character

6.116.2.78 `char * brathl::CTools::ToUpper (char * str) [static]`

Set a string in uppercase

Parameters

<i>str</i>	[in/out] : string to be modified
------------	----------------------------------

Returns

a pointer to the string

Referenced by StringToUpper().

6.116.2.79 `char brathl::CTools::ToUpper (const char chr) [static]`

Set a character in uppercase

Parameters

<i>in</i>	<i>chr</i> : character
-----------	------------------------

Returns

the uppercase character

6.116.2.80 `string brathl::CTools::TrailingZeroesTrim (const string & Text, bool dotTrim = true) [static]`

Removes trailing zeroes from a number: 2.30000 is transformed into 2.3.

Parameters

<i>in</i>	<i>Text</i> : String
<i>in</i>	<i>dotTrim</i> : if true, remove dot at the end : 2.000 -> 2, if false, leave dot : 2.000 -> 2.

Returns

Returns modified string

6.116.2.81 `char * brathl::CTools::Trim (char * str) [static]`

Remove all the blank characters at the beginning and the end of a string. Blank characters are identified by the function isspace (3C).

Parameters

<i>str</i>	[in/out] : string to be modified
------------	----------------------------------

Returns

a pointer to the string

References StringTrim().

Referenced by brathl::CFile::ReadLineData().

6.116.2.82 `double brathl::CTools::UnaryMinus (double X) [static]`

Negates a number. Takes default values into account

Parameters

<i>in</i>	<i>X</i>	: Number involved
-----------	----------	-------------------

Returns

Negated number

6.116.2.83 `double brathl::CTools::UnaryNot (double X) [static]`

Negates a logical value (0 is false, other (except default value) is true. Takes default values into account

Parameters

<i>in</i>	<i>X</i>	: Number involved
-----------	----------	-------------------

Returns

Negated value

6.116.2.84 `double brathl::CTools::UnconvertLat (const string & value) [static]`

Converts and normalize a latitude string representation (eg 60 N, 75.56 W, 60, -75.56) Normalize +/-90.

Parameters

<i>value</i>	latitude string representation
--------------	--------------------------------

References StringTrim(), and StrToDouble().

6.116.2.85 `double brathl::CTools::UnconvertLon (const string & value, bool normalize = true) [static]`

Converts and eventually normalize a longitude string representation (eg 60 E, 120.23 W, 60, -120.23) Normalize +/-180.

Parameters

<i>normalize</i>	set to true to normalize longitude value
<i>value</i>	longitude string representation

Returns

converted longitude.

References StringTrim(), and StrToDouble().

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

- Tools.h
- Tools.cpp

6.117 brathl::CTreeField Class Reference

```
#include <TreeField.h>
```

Inherits brathl::CObjectTree.

Public Member Functions

- virtual CObjectTreeIterator **AddChild** (CObjectTreeNode *parent, const string &nm, **CField** *x, bool goCurrent=false)
- virtual CObjectTreeIterator **AddChild** (CObjectTreeIterator &parent, const string &nm, **CField** *x, bool goCurrent=false)
- virtual CObjectTreeIterator **AddChild** (const string &nm, **CField** *x, bool goCurrent=false)
- **CTreeField** ()
Empty CTreeField (p. 347) ctor.
- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- void **DumpDictionary** (ostream &fOut=cout)
- void **DumpDictionary** (const string &outputFileName)
- **CField** * **FindParent** (**CField** *field)
- **CField** * **GetCurrentData** (bool withExcept=true)
- **CField** * **GetParentData** (bool withExcept=true)
- **CField** * **GetRootData** ()
- void **ResetHiddenFlag** ()
- virtual ~**CTreeField** ()
Destructor.

Static Public Member Functions

- static **CField** * **GetDataAsFieldObject** (CObjectTreeNode *node, bool withExcept=true)
- static **CFieldRecord** * **GetDataAsFieldRecordObject** (CObjectTreeNode *node, bool withExcept=true)

Static Public Attributes

- static const string **m_keyDelimiter** = "."

Additional Inherited Members

6.117.1 Detailed Description

Tree fields management class.

Version

1.0

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

- TreeField.h
- TreeField.cpp

6.118 brathl::CUInt16Array Class Reference

```
#include <List.h>
```


Public Member Functions

- virtual bool **Complement** (const **CUInt16Array** &array, **CUInt16Array** &complement) const
- **CUInt16Array** ()
Empty CUInt16Array (p. 348) ctor.
- **CUInt16Array** (const **CUInt16Array** &vect)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CUInt16Array::iterator it)
- virtual void **Insert** (**CUInt16Array** *vect, bool bEnd=true)
- virtual void **Insert** (const **CUInt16Array** &vect, bool bEnd=true)
- virtual void **Insert** (const vector< uint16_t > &vect, bool bEnd=true)
- virtual void **Insert** (uint16_t *vect, size_t length)
- virtual void **Insert** (const uint16_t value)
- virtual CUInt16Array::iterator **InsertAt** (CUInt16Array::iterator where, const uint16_t value)
- virtual CUInt16Array::iterator **InsertAt** (int32_t index, const uint16_t value)
- virtual bool **Intersect** (const **CUInt16Array** &array, **CUInt16Array** &intersect) const
- virtual bool **operator!=** (const **CUInt16Array** &vect)
- virtual const **CUInt16Array** & **operator=** (const **CUInt16Array** &vect)
- virtual bool **operator==** (const **CUInt16Array** &vect)
- virtual void **RemoveAll** ()
- virtual CUInt16Array::iterator **ReplaceAt** (CUInt16Array::iterator where, const uint16_t value)
- virtual CUInt16Array::iterator **ReplaceAt** (int32_t index, const uint16_t value)
- virtual uint16_t * **ToArray** ()
- virtual int16_t * **ToIntArray** ()
- virtual string **ToString** (const string &delim="," , bool useBracket=true) const
- virtual ~**CUInt16Array** ()
Destructor.

6.118.1 Detailed Description

An array (vector) of ints management class.

Version

1.0

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

- List.h
- List.cpp

6.119 brathl::CUInt64Array Class Reference

```
#include <List.h>
```

Public Member Functions

- **CUInt64Array** ()
Empty CUIntArray (p. 351) ctor.
- **CUInt64Array** (const **CUInt64Array** &vect)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.

- virtual bool **Erase** (CUInt64Array::iterator it)
- uint64_t **GetProductValues** () const
- virtual void **Insert** (CUInt64Array *vect, bool bEnd=true)
- virtual void **Insert** (const CUInt64Array &vect, bool bEnd=true)
- virtual void **Insert** (const vector< uint64_t > &vect, bool bEnd=true)
- virtual void **Insert** (uint64_t *vect, size_t length)
- virtual void **Insert** (const uint64_t value)
- virtual CUInt64Array::iterator **InsertAt** (CUInt64Array::iterator where, const uint64_t value)
- virtual CUInt64Array::iterator **InsertAt** (size_t index, const uint64_t value)
- virtual bool **Intersect** (const CUInt64Array &array, CUInt64Array &intersect) const
- virtual bool **operator!=** (const CUInt64Array &vect)
- virtual const CUInt64Array & **operator=** (const CUInt64Array &vect)
- virtual bool **operator==** (const CUInt64Array &vect)
- virtual void **RemoveAll** ()
- virtual CUInt64Array::iterator **ReplaceAt** (CUInt64Array::iterator where, const uint64_t value)
- virtual CUInt64Array::iterator **ReplaceAt** (size_t index, const uint64_t value)
- virtual uint64_t * **ToArray** ()
- virtual string **Tostring** (const string &delim="," , bool useBracket=true) const
- virtual ~CUInt64Array ()

Destructor.

6.119.1 Detailed Description

An array (vector) of ints management class.

Version

1.0

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

- List.h
- List.cpp

6.120 brathl::CUInt8Array Class Reference

```
#include <List.h>
```

Public Member Functions

- virtual bool **Complement** (const CUInt8Array &array, CUInt8Array &complement) const
- CUInt8Array ()
Empty CUInt8Array (p. 350) ctor.
- CUInt8Array (const CUInt8Array &vect)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CUInt8Array::iterator it)
- virtual void **Insert** (CUInt8Array *vect, bool bEnd=true)
- virtual void **Insert** (const CUInt8Array &vect, bool bEnd=true)
- virtual void **Insert** (const vector< uint8_t > &vect, bool bEnd=true)
- virtual void **Insert** (uint8_t *vect, size_t length)
- virtual void **Insert** (const uint8_t value)
- virtual CUInt8Array::iterator **InsertAt** (CUInt8Array::iterator where, const uint8_t value)

- virtual CUIntArray::iterator **InsertAt** (int32_t index, const uint8_t value)
- virtual bool **Intersect** (const **CUIntArray** &array, **CUIntArray** &intersect) const
- virtual bool **operator!=** (const **CUIntArray** &vect)
- virtual const **CUIntArray** & **operator=** (const **CUIntArray** &vect)
- virtual bool **operator==** (const **CUIntArray** &vect)
- virtual void **RemoveAll** ()
- virtual CUIntArray::iterator **ReplaceAt** (CUIntArray::iterator where, const uint8_t value)
- virtual CUIntArray::iterator **ReplaceAt** (int32_t index, const uint8_t value)
- virtual uint8_t * **ToArray** ()
- virtual int8_t * **ToIntArray** ()
- virtual string **Tostring** (const string &delim=",", bool useBracket=true) const
- virtual ~**CUIntArray** ()

Destructor.

6.120.1 Detailed Description

An array (vector) of ints management class.

Version

1.0

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

- List.h
- List.cpp

6.121 brathl::CUIntArray Class Reference

```
#include <List.h>
```

Public Member Functions

- virtual bool **Complement** (const **CUIntArray** &array, **CUIntArray** &complement) const
- **CUIntArray** ()
*Empty **CUIntArray** (p. 351) ctor.*
- **CUIntArray** (const **CUIntArray** &vect)
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CUIntArray::iterator it)
- uint32_t **GetProductValues** () const
- virtual void **Insert** (**CUIntArray** *vect, bool bEnd=true)
- virtual void **Insert** (const **CUIntArray** &vect, bool bEnd=true)
- virtual void **Insert** (const vector< uint32_t > &vect, bool bEnd=true)
- virtual void **Insert** (uint32_t *vect, size_t length)
- virtual void **Insert** (const uint32_t value)
- virtual CUIntArray::iterator **InsertAt** (CUIntArray::iterator where, const uint32_t value)
- virtual CUIntArray::iterator **InsertAt** (int32_t index, const uint32_t value)
- virtual bool **Intersect** (const **CUIntArray** &array, **CUIntArray** &intersect) const
- virtual bool **operator!=** (const **CUIntArray** &vect)
- virtual const **CUIntArray** & **operator=** (const **CUIntArray** &vect)
- virtual bool **operator==** (const **CUIntArray** &vect)
- virtual void **RemoveAll** ()

- virtual CUIntArray::iterator **ReplaceAt** (CUIntArray::iterator where, const uint32_t value)
- virtual CUIntArray::iterator **ReplaceAt** (int32_t index, const uint32_t value)
- virtual uint32_t * **ToArray** ()
- virtual int32_t * **ToIntArray** ()
- virtual size_t * **ToSizeTArray** ()
- virtual string **ToString** (const string &delim="," , bool useBracket=true) const
- virtual ~CUIntArray ()

Destructor.

6.121.1 Detailed Description

An array (vector) of ints management class.

Version

1.0

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

- List.h
- List.cpp

6.122 brathl::CUIntArray Class Reference

```
#include <List.h>
```

Inherited by CMapDataMode, CMapProjection, CMapTypeData, CMapTypeDisp, CMapTypeField, CMapTypeFilter, and CMapTypeOp.

Public Member Functions

- **CUIntArray** ()
CUIntArray (p. 352) ctor.
- virtual void **Dump** (ostream &fOut=cerr) const
Dump fonction.
- virtual bool **Erase** (CUIntArray::iterator it)
- virtual bool **Erase** (const string &key)
- virtual uint32_t **Exists** (const string &key) const
- virtual void **GetKeys** (CStringArray &keys, bool bRemoveAll=true)
- virtual uint32_t **Insert** (const string &key, uint32_t value, bool withExcept=true)
- virtual void **Insert** (const **CUIntArray** &m, bool bRemoveAll=true, bool withExcept=true)
- virtual void **Insert** (const CStringArray &keys, uint32_t initValue, bool bRemoveAll=true, bool withExcept=true)
- virtual void **Insert** (const CStringArray &keys, const **CUIntArray** &values, bool bRemoveAll=true, bool withExcept=true)
- virtual void **Insert** (const CStringArray &keys, bool bRemoveAll=true, bool withExcept=true)
- virtual uint32_t **operator[]** (const string &key)
- virtual void **RemoveAll** ()
- virtual ~CUIntArray ()
CUIntArray (p. 352) dtor.

6.122.1 Detailed Description

a set of unsigned integer value management classes.

Version

1.0

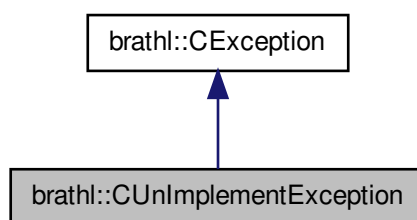
The documentation for this class was generated from the following files:

- List.h
- List.cpp

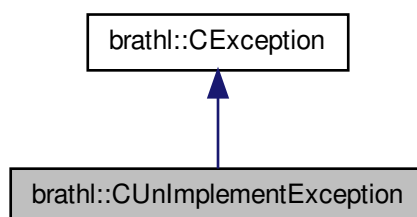
6.123 bratl::CUnImplementException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for bratl::CUnImplementException:



Collaboration diagram for bratl::CUnImplementException:



Public Member Functions

- **CUnImplementException** ()
Empty CUnImplementException (p. 353) ctor.
- virtual const char * **TypeOf** () const

Identification of exception (human readable)

- virtual **~CUnImplementException** () throw ()

Destructor.

- **CUnImplementException** (const string &message, int32_t errcode=**BRATHL_UNIMPLEMENT_ERROR**)

Additional Inherited Members

6.123.1 Detailed Description

Unimplement Exception management class.

Version

1.0

6.123.2 Constructor & Destructor Documentation

6.123.2.1 **brathl::CUnImplementException::CUnImplementException** (const string & *message*, int32_t *errcode* = **BRATHL_UNIMPLEMENT_ERROR**) *[inline]*

Creates a new **CUnImplementException** (p. 353) object.

Parameters

<i>message</i>	<i>[in]</i> : error message
<i>errcode</i>	<i>[in]</i> : error code

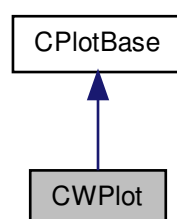
The documentation for this class was generated from the following file:

- **Exception.h**

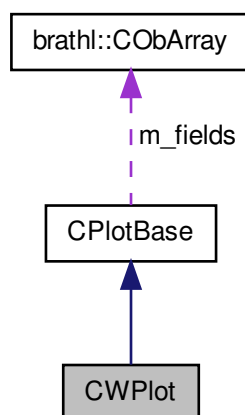
6.124 CWPlot Class Reference

```
#include <WPlot.h>
```

Inheritance diagram for CWPlot:



Collaboration diagram for CWPlot:



Public Member Functions

- **CWPlot** (uint32_t groupNumber=0)
- virtual void **GetInfo** ()
- virtual **CInternalFiles** * **GetInternalFiles** (CBratObject *ob, bool withExcept=true)

Static Public Member Functions

- static **CInternalFilesZFX** * **GetInternalFilesZFX** (CBratObject *ob, bool withExcept=true)

Protected Member Functions

- void **Init** ()

Additional Inherited Members

6.124.1 Detailed Description

A **CWPlot** (p. 354) object management class

Version

1.0

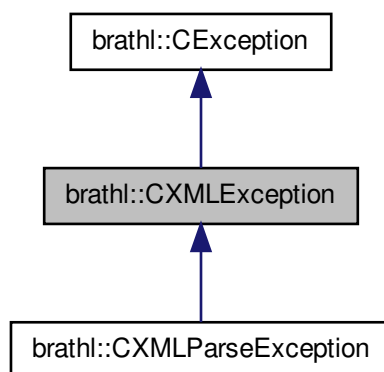
The documentation for this class was generated from the following files:

- WPlot.h
- WPlot.cpp

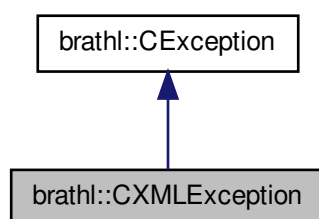
6.125 bratl::CXMLException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for bratl::CXMLException:



Collaboration diagram for bratl::CXMLException:



Public Member Functions

- **CXMLException** ()
Empty CParameterException (p. 278) ctor.
- **CXMLException** (const string &message, int32_t errcode)
- virtual const char * **TypeOf** () const
Identification of exception (human readable)
- virtual ~**CXMLException** () throw ()
Destructor.

Additional Inherited Members

6.125.1 Detailed Description

XML Exception management class.

Version

1.0

6.125.2 Constructor & Destructor Documentation

6.125.2.1 bratl::CXMLException::CXMLException (const string & *message*, int32_t *errcode*) [inline]

Creates a new **CParameterException** (p. 278) object.

Parameters

<i>message</i>	[in] : error message
<i>errcode</i>	[in] : error code

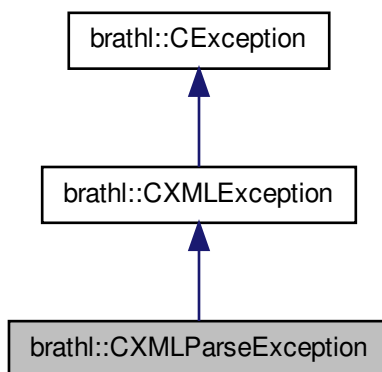
The documentation for this class was generated from the following file:

- **Exception.h**

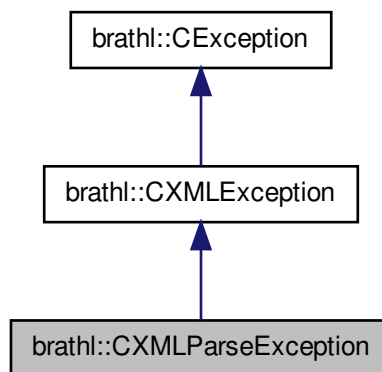
6.126 bratl::CXMLParseException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for bratl::CXMLParseException:



Collaboration diagram for bratl::CXMLParseException:



Public Member Functions

- **CXMLParseException** (const string &message, int32_t errcode)
- virtual const char * **TypeOf** () const
Identification of exception (human readable)
- virtual ~**CXMLParseException** () throw ()
Destructor.

Additional Inherited Members

6.126.1 Detailed Description

XML Parse Exception management class.

Version

1.0

6.126.2 Constructor & Destructor Documentation

6.126.2.1 `bratl::CXMLParseException::CXMLParseException (const string & message, int32_t errcode) [inline]`

Creates a new **CParameterException** (p. 278) object.

Parameters

<i>message</i>	[in] : error message
<i>errcode</i>	[in] : error code

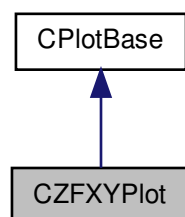
The documentation for this class was generated from the following file:

- **Exception.h**

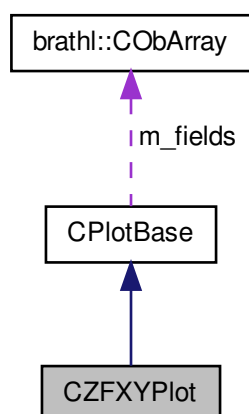
6.127 CZFXYPlot Class Reference

```
#include <ZFXYPlot.h>
```

Inheritance diagram for CZFXYPlot:



Collaboration diagram for CZFXYPlot:



Public Member Functions

- **CZFXYPlot** (uint32_t groupNumber=0)
- virtual void **GetInfo** ()
- virtual **CInternalFiles** * **GetInternalFiles** (CBratObject *ob, bool withExcept=true)
- void **GetPlotWidthHeight** (**CInternalFiles** *zfx, const string &fieldName, int32_t &width, int32_t &height, **CExpressionValue** &varY, **CExpressionValue** &varX, uint32_t &dimRangeX, uint32_t &dimRangeY, string &varXName, string &varYName)

Static Public Member Functions

- static **CInternalFilesYFX** * **GetInternalFilesYFX** (CBratObject *ob, bool withExcept=true)
- static **CInternalFilesZFX** * **GetInternalFilesZFX** (CBratObject *ob, bool withExcept=true)

Protected Member Functions

- void **Init** ()

Additional Inherited Members

6.127.1 Detailed Description

A **CZFXYPLOT** (p. 359) object management class

Version

1.0

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

- ZFXYPLOT.h
- ZFXYPLOT.cpp

6.128 vtkObArray Class Reference

```
#include <vtkList.h>
```

Public Member Functions

- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- virtual bool **Erase** (vtkObArray::iterator it)
- virtual bool **Erase** (int32_t index)
- bool **GetDelete** ()
- virtual void **Insert** (vtkObject *ob)
- virtual vtkObArray::iterator **InsertAt** (vtkObArray::iterator where, vtkObject *ob)
- virtual bool **PopBack** ()
- virtual void **RemoveAll** ()
- virtual vtkObArray::iterator **ReplaceAt** (vtkObArray::iterator where, vtkObject *ob)
- void **SetDelete** (bool value)
- **vtkObArray** (bool bDelete=true)
Empty vtkObArray (p. 360) ctor.
- virtual ~**vtkObArray** ()
Destructor.

Protected Attributes

- bool **m_bDelete**

6.128.1 Detailed Description

An array (vector) of vtkObject management class.

Version

1.0

6.128.2 Constructor & Destructor Documentation

6.128.2.1 vtkObArray::~vtkObArray () [virtual]

Destructor.

Creates new vtkObArray object from another vtkObArray

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

References RemoveAll().

6.128.3 Member Function Documentation

6.128.3.1 bool vtkObArray::Erase (vtkObArray::iterator *it*) [virtual]

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by Erase().

6.128.3.2 bool vtkObArray::Erase (int32_t *index*) [virtual]

Delete an element referenced by it

Returns

true if no error, otherwise false

References Erase().

6.128.3.3 bool vtkObArray::GetDelete () [inline]

Copy a new **vtkObArray** (p. 360) to the object

6.128.3.4 void vtkObArray::RemoveAll () [virtual]

Remove all elements and clear the list

Referenced by ~vtkObArray().

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

- vtkList.h
- vtkList.cpp

6.129 vtkObList Class Reference

```
#include <vtkList.h>
```

Public Member Functions

- virtual void **Dump** (ostream &fOut=cerr)

Dump function.

- virtual bool **Erase** (vtkObList::iterator it)
- bool **GetDelete** ()
- virtual void **Insert** (vtkObject *ob, bool bEnd=true)
- virtual bool **PopBack** ()
- virtual void **RemoveAll** ()
- void **SetDelete** (bool value)
- **vtkObList** (bool bDelete=true)

*Empty **vtkObList** (p. 361) ctor.*

- virtual **~vtkObList** ()

Destructor.

Protected Attributes

- bool **m_bDelete**

6.129.1 Detailed Description

A list of vtkObject management class.

Version

1.0

6.129.2 Constructor & Destructor Documentation

6.129.2.1 **vtkObList::~vtkObList** () [virtual]

Destructor.

Creates new vtkObList object from another CStringList

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

References RemoveAll().

6.129.3 Member Function Documentation

6.129.3.1 **bool vtkObList::Erase** (vtkObList::iterator *it*) [virtual]

Delete an element referenced by iteratorMnemo

Returns

true if no error, otherwise false

6.129.3.2 **bool vtkObList::GetDelete** () [inline]

Copy a new CStringList to the object

6.129.3.3 **void vtkObList::RemoveAll** () [virtual]

Remove all elements and clear the list

Referenced by ~vtkObList().

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

- vtkList.h
- vtkList.cpp

6.130 vtkObMap Class Reference

```
#include <vtkList.h>
```

Public Member Functions

- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- bool **Erase** (vtkObMap::iterator it)
- bool **Erase** (const string &key)
- vtkObject * **Exists** (const string &key)
- bool **GetDelete** ()
- vtkObject * **Insert** (const string &key, vtkObject *ob, bool withExcept=true)
- vtkObject * **operator[]** (const string &key)
- void **RemoveAll** ()
- void **SetDelete** (bool value)
- **vtkObMap** (bool bDelete=true)
vtkObMap (p. 363) ctor
- virtual ~**vtkObMap** ()
vtkObMap (p. 363) dtor

Protected Attributes

- bool **m_bDelete**

6.130.1 Detailed Description

a set of object management classes.

Version

1.0

6.130.2 Member Function Documentation

6.130.2.1 bool vtkObMap::Erase (vtkObMap::iterator it)

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by Erase().

6.130.2.2 bool vtkObMap::Erase (const string & key)

Delete an element by its key

Returns

true if no error, otherwise false

References Erase().

6.130.2.3 vtkObject * vtkObMap::Exists (const string & key)

Inserts a **vtkObMap** (p. 363)

Parameters

<i>obMap</i>	: vtkObMap (p. 363) to insert
<i>withExcept</i>	: true for exception handling, flse otherwise Tests if an element identify by 'key' already exists

Returns

a vtkObject pointer if exists, otherwise NULL

6.130.2.4 vtkObject * vtkObMap::Insert (const string & key, vtkObject * ob, bool withExcept = true)

Inserts a vtkObject object

Parameters

<i>key</i>	: vtkObject name (map key)
<i>ob</i>	: vtkObject value
<i>withExcept</i>	: true for exception handling, flse otherwise

Returns

vtkObject object or NULL if error

References BRATHL_LOGIC_ERROR.

6.130.2.5 vtkObject * vtkObMap::operator[] (const string & key)

operator[] redefinition. Searches a vtkObject object identify by 'key'.

Parameters

<i>key</i>	: vtkObject keyword
------------	---------------------

Returns

a pointer to the vtkObject object if found, NULL if not found

6.130.2.6 void vtkObMap::RemoveAll ()

Remove all elements and clear the map

Referenced by ~vtkObMap().

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

- vtkList.h

- vtkList.cpp

6.131 wxObArray Class Reference

```
#include <wxList.h>
```

Public Member Functions

- virtual void **Dump** (ostream &fOut=cerr)
Dump function.
- virtual bool **Erase** (wxObArray::iterator it)
- virtual bool **Erase** (int32_t index)
- bool **GetDelete** ()
- virtual void **Insert** (wxObject *ob)
- virtual wxObArray::iterator **InsertAt** (wxObArray::iterator where, wxObject *ob)
- virtual bool **PopBack** ()
- virtual void **RemoveAll** ()
- virtual wxObArray::iterator **ReplaceAt** (wxObArray::iterator where, wxObject *ob)
- void **SetDelete** (bool value)
- **wxObArray** (bool bDelete=true)
Empty wxObArray (p. 365) ctor.
- virtual ~**wxObArray** ()
Destructor.

Protected Attributes

- bool **m_bDelete**

6.131.1 Detailed Description

An array (vector) of wxObject management class.

Version

1.0

6.131.2 Constructor & Destructor Documentation

6.131.2.1 wxObArray::~wxObArray () [virtual]

Destructor.

Creates new **wxObArray** (p. 365) object from another **wxObArray** (p. 365)

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

References RemoveAll().

6.131.3 Member Function Documentation

6.131.3.1 `bool wxObArray::Erase (wxObArray::iterator it) [virtual]`

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by `Erase()`.

6.131.3.2 `bool wxObArray::Erase (int32_t index) [virtual]`

Delete an element referenced by it

Returns

true if no error, otherwise false

References `Erase()`.

6.131.3.3 `bool wxObArray::GetDelete () [inline]`

Copy a new **wxObArray** (p. 365) to the object

6.131.3.4 `void wxObArray::RemoveAll () [virtual]`

Remove all elements and clear the list

Referenced by `~wxObArray()`.

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

- wxList.h
- wxList.cpp

6.132 wxObList Class Reference

```
#include <wxList.h>
```

Public Member Functions

- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- virtual bool **Erase** (wxObList::iterator it)
- bool **GetDelete** ()
- virtual void **Insert** (wxObject *ob, bool bEnd=true)
- virtual bool **PopBack** ()
- virtual void **RemoveAll** ()
- void **SetDelete** (bool value)
- **wxObList** (bool bDelete=true)
Empty wxObList (p. 366) ctor.
- virtual **~wxObList** ()
Destructor.

Protected Attributes

- bool **m_bDelete**

6.132.1 Detailed Description

A list of wxObject management class.

Version

1.0

6.132.2 Constructor & Destructor Documentation

6.132.2.1 wxObList::~wxObList () [virtual]

Destructor.

Creates new **wxObList** (p. 366) object from another CStringList

Parameters

<i>list</i>	[in] : list to be copied
-------------	--------------------------

References RemoveAll().

6.132.3 Member Function Documentation

6.132.3.1 bool wxObList::Erase (wxObList::iterator it) [virtual]

Delete an element referenced by iteratorMnemo

Returns

true if no error, otherwise false

6.132.3.2 bool wxObList::GetDelete () [inline]

Copy a new CStringList to the object

6.132.3.3 void wxObList::RemoveAll () [virtual]

Remove all elements and clear the list

Referenced by ~wxObList().

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

- wxList.h
- wxList.cpp

6.133 wxObMap Class Reference

```
#include <wxList.h>
```

Inherited by CMapProcess.

Public Member Functions

- virtual void **Dump** (ostream &fOut=cerr)
Dump fonction.
- bool **Erase** (wxObMap::iterator it)

- bool **Erase** (const string &key)
- wxObject * **Exists** (const string &key)
- bool **GetDelete** ()
- wxObject * **Insert** (const string &key, wxObject *ob, bool withExcept=true)
- wxObject * **operator[]** (const string &key)
- void **RemoveAll** ()
- void **SetDelete** (bool value)
- **wxObMap** (bool bDelete=true)
 - wxObMap* (p. 367) ctor
- virtual ~**wxObMap** ()
 - wxObMap* (p. 367) dtor

Protected Attributes

- bool **m_bDelete**

6.133.1 Detailed Description

a set of object management classes.

Version

1.0

6.133.2 Member Function Documentation

6.133.2.1 bool wxObMap::Erase (wxObMap::iterator it)

Delete an element referenced by it

Returns

true if no error, otherwise false

Referenced by Erase().

6.133.2.2 bool wxObMap::Erase (const string & key)

Delete an element by its key

Returns

true if no error, otherwise false

References Erase().

6.133.2.3 wxObject * wxObMap::Exists (const string & key)

Inserts a **wxObMap** (p. 367)

Parameters

<i>obMap</i>	: wxObMap (p. 367) to insert
<i>withExcept</i>	: true for exception handling, flse otherwise Tests if an element identify by 'key' already exists

Returns

a wxObject pointer if exists, otherwise NULL

6.133.2.4 `wxObject * wxObMap::Insert (const string & key, wxObject * ob, bool withExcept = true)`

Inserts a wxObject object

Parameters

<i>key</i>	: wxObject name (map key)
<i>value</i>	: wxObject value
<i>withExcept</i>	: true for exception handling, flse otherwise

Returns

wxObject object or NULL if error

References BRATHL_LOGIC_ERROR.

6.133.2.5 `wxObject * wxObMap::operator[] (const string & key)`

operator[] redefinition. Searches a wxObject object identifiy by 'key'.

Parameters

<i>key</i>	: wxObject keyword
------------	--------------------

Returns

a pointer to the wxObject object if found, NULL if not found

6.133.2.6 `void wxObMap::RemoveAll ()`

Remove all elements and clear the map

Referenced by ~wxObMap().

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

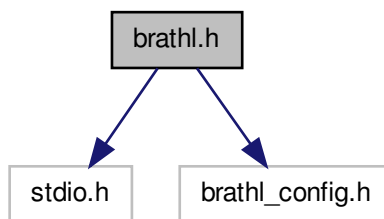
- wxList.h
- wxList.cpp

7 File Documentation

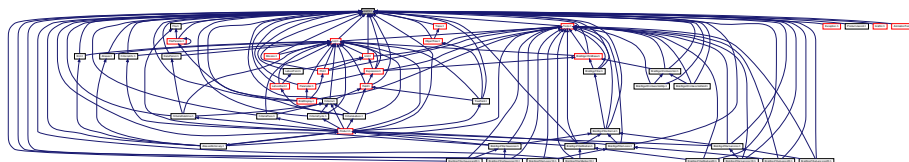
7.1 brathl.h File Reference

```
#include <stdio.h>
#include "brathl_config.h"
```

Include dependency graph for brathl.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct **_structDateDSM**
- struct **_structDateJulian**
- struct **_structDateSecond**
- struct **_structDateYMDHMSM**

Macros

- #define **__attribute__(x)**
- #define **BRATHL_CYCLE_LEN** 60
- #define **BRATHL_MAX_ERRMSG_LEN** 255
- #define **BRATHL_PATH_MAX** PATH_MAX
- #define **BRATHL_REF_DATE_USER_LEN** 28
- #define **LIBRATHL_API**
- #define **M_PI** 3.14159265358979323846
- #define **M_PI_2** 1.57079632679489661923 /* pi/2 */
- #define **M_PI_4** 0.78539816339744830962 /* pi/4 */

Typedefs

- typedef struct **_structDateDSM** brathl_DateDSM
- typedef struct **_structDateJulian** brathl_DateJulian
- typedef struct **_structDateSecond** brathl_DateSecond
- typedef struct **_structDateYMDHMSM** brathl_DateYMDHMSM

Enumerations

- enum **brathl_FileMode** { **ReadOnly**, **Write**, **Replace**, **New** }
- enum **brathl_global_constants** {
EARTH_ROTATION = 0, **LIGHT_SPEED**, **EARTH_GRAVITY**, **EARTH_RADIUS**,
ELLIPSOID_PARAM }
- enum **brathl_mission** {
TOPEX, **JASON2**, **JASON1**, **ERS2**,
ENVISAT, **ERS1_A**, **ERS1_B**, **GFO** }
- enum **brathl_refDate** {
REF19500101, **REF19580101**, **REF19850101**, **REF19900101**,
REF20000101, **REFUSER1**, **REFUSER2** }

Variables

- LIBRATHL_API char **brathl_refDateUser1** [**BRATHL_REF_DATE_USER_LEN**]
- LIBRATHL_API char **brathl_refDateUser2** [**BRATHL_REF_DATE_USER_LEN**]

7.1.1 Detailed Description

C/C++ general interface of BRATHL

7.1.2 Macro Definition Documentation

7.1.2.1 #define BRATHL_CYCLE_LEN 60

Maximum length of date reference string

7.1.2.2 #define BRATHL_MAX_ERRMSG_LEN 255

Maximum length of error message string

7.1.2.3 #define BRATHL_REF_DATE_USER_LEN 28

Maximum length of date reference string

7.1.3 Typedef Documentation

7.1.3.1 typedef struct _structDateDSM brathl_DateDSM

Day/seconds/microseconds date structureCreates a type name for **_structDateDSM** (p. 97)

7.1.3.2 typedef struct _structDateJulian brathl_DateJulian

Decimal julian date structureCreates a type name for **_structDateJulian** (p. 97)

7.1.3.3 typedef struct _structDateSecond brathl_DateSecond

Decimal seconds date structureCreates a type name for **_structDateSecond** (p. 98)

7.1.3.4 typedef struct _structDateYMDHMSM brathl_DateYMDHMSM

YYYY-MM-DD HH:MN:SS:MS date structureCreates a type name for **_structDateYMDHMSM** (p. 99)

7.1.4 Enumeration Type Documentation

7.1.4.1 enum brathl_FileMode

Enumerator:

Write file exists, open read-only

Replace file exists, open for writing

New create new file, even if it already exists create new file, fail if it already exists

7.1.4.2 enum brathl_mission

Satellite (mission) enumeration

Enumerator:

TOPEX Topex/Poseidon

JASON2 Jason-2

JASON1 Jason-1

ERS2 ERS2

ENVISAT Envisat

ERS1_A ERS1-A

ERS1_B ERS1-B

GFO GFO

7.1.4.3 enum brathl_refDate

date reference enumeration Used to give a date a a start reference User can defined its own reference by using REFUSER1 and/or REFUSER2

Enumerator:

REF19500101 reference to the 1950-01-01 00:00:00:00

REF19580101 reference to the 1958-01-01 00:00:00:00

REF19850101 reference to the 1985-01-01 00:00:00:00

REF19900101 reference to the 1990-01-01 00:00:00:00

REF20000101 reference to the 2000-01-01 00:00:00:00

REFUSER1 reference to a user-defined date **brathl_refDateUser1** (p. 372)

REFUSER2 reference to a second user-defined date **brathl_refDateUser2** (p. 372)

7.1.5 Variable Documentation

7.1.5.1 LIBRATHL_API char brathl_refDateUser1[BRATHL_REF_DATE_USER_LEN]

Global variable to define REFUSER1 date (see **brathl_refDate** (p. 372))

Referenced by brathl::CDate::ConstructDate().

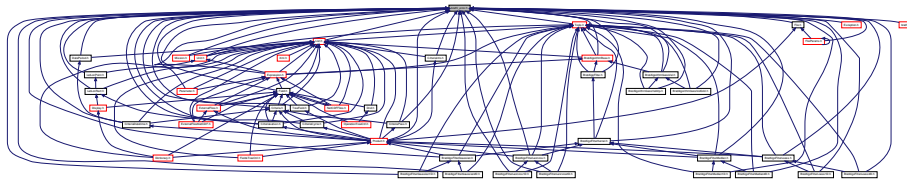
7.1.5.2 LIBRATHL_API char brathl_refDateUser2[BRATHL_REF_DATE_USER_LEN]

Global variable to define REFUSER2 date (see **brathl_refDate** (p. 372))

Referenced by brathl::CDate::ConstructDate().

7.2 brathl_error.h File Reference

This graph shows which files directly or indirectly include this file:



Macros

- **#define BRATHL_COUNT_ERROR -4**
Count error.
- **#define BRATHL_ERROR -1**
General error.
- **#define BRATHL_ERROR_INVALID_DATE -101**
Invalid date.
- **#define BRATHL_ERROR_INVALID_DATE_NEGATIVE -112**
Invalid date (date must be > 0)
- **#define BRATHL_ERROR_INVALID_DATE_REF -102**
Invalid reference date.
- **#define BRATHL_ERROR_INVALID_DATE_REF_CONV -103**
Invalid reference date conversion.
- **#define BRATHL_ERROR_INVALID_DAY -107**
Invalid day value.
- **#define BRATHL_ERROR_INVALID_DSM -104**
Invalid days or seconds or museonds values (must be > 0)
- **#define BRATHL_ERROR_INVALID_HOUR -108**
Invalid hour value (must be >= 0 and <= 23)
- **#define BRATHL_ERROR_INVALID_MINUTE -109**
Invalid minute value (must be >= 0 and <= 59)
- **#define BRATHL_ERROR_INVALID_MISSION -203**
Unknown mission value.
- **#define BRATHL_ERROR_INVALID_MONTH -106**
Invalid month value (must be >= 1 and <= 12)
- **#define BRATHL_ERROR_INVALID_MUSECOND -111**
Invalid musecond value (must be >= 0 and <= 999999)
- **#define BRATHL_ERROR_INVALID_NB_PASS -201**
Invalid nb pass (must be > 0)
- **#define BRATHL_ERROR_INVALID_REPETITION -202**
Invalid repetition (must be > 0)
- **#define BRATHL_ERROR_INVALID_SECOND -110**
Invalid second value (must be >= 0 and <= 59)
- **#define BRATHL_ERROR_INVALID_YEAR -105**
Invalid year value (must be >= 1950)
- **#define BRATHL_INCONSISTENCY_ERROR -11**
Inconsistency error.
- **#define BRATHL_IO_ERROR -7**
I/O error.

- **#define BRATHL_LIMIT_ERROR** -6
Limit error.
- **#define BRATHL_LOGIC_ERROR** -10
Logic error (program error)
- **#define BRATHL_MEMORY_ERROR** -8
Memory error.
- **#define BRATHL_RANGE_ERROR** -5
Range error.
- **#define BRATHL_SUCCESS** 0
- **#define BRATHL_SYNTAX_ERROR** -2
Syntax error.
- **#define BRATHL_SYSTEM_ERROR** -9
System error.
- **#define BRATHL_UNIMPLEMENT_ERROR** -12
error for non non implement code
- **#define BRATHL_UNIT_ERROR** -3
Unit error.
- **#define BRATHL_WARNING** 2
warning
- **#define BRATHL_WARNING_INVALID_REF_FILE_FIELD** -205
WARNING - Invalid reference mission file format.
- **#define BRATHL_WARNING_INVALID_REF_FILE_FIELDDATE** -206
WARNING - Invalid reference mission date.
- **#define BRATHL_WARNING_OPEN_FILE_ALIAS_MISSION** -207
WARNING - Unable to open alias mission file.
- **#define BRATHL_WARNING_OPEN_FILE_REF_FILE** -204
WARNING - Unable to open reference mission file.
- **#define LIBRATHL_API**

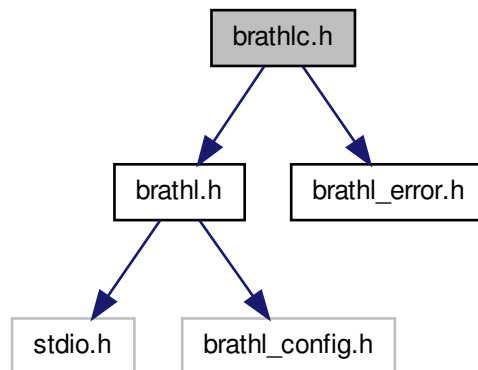
7.2.1 Detailed Description

BRATHL error codes

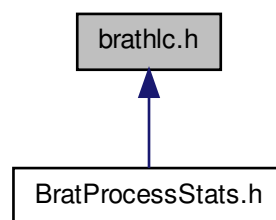
7.3 brathlc.h File Reference

```
#include "brathl.h"
#include "brathl_error.h"
```

Include dependency graph for brathlc.h:



This graph shows which files directly or indirectly include this file:



Functions

- LIBRATHL_API int32_t **brathl_Cycle2YMDHMSM** (**brathl_mission** mission, uint32_t cycle, uint32_t pass, **brathl_DateYMDHMSM** *dateYMDHMSM)
- LIBRATHL_API int32_t **brathl_DayOfYear** (**brathl_DateYMDHMSM** *dateYMDHMSM, uint32_t *dayOfYear)
- LIBRATHL_API int32_t **brathl_DiffDSM** (**brathl_DateDSM** *dateDSM1, **brathl_DateDSM** *dateDSM2, double *diff)
- LIBRATHL_API int32_t **brathl_DiffJulian** (**brathl_DateJulian** *dateJulian1, **brathl_DateJulian** *dateJulian2, double *diff)
- LIBRATHL_API int32_t **brathl_DiffYMDHMSM** (**brathl_DateYMDHMSM** *dateYMDHMSM1, **brathl_DateYMDHMSM** *dateYMDHMSM2, double *diff)
- LIBRATHL_API int32_t **brathl_DSM2Julian** (**brathl_DateDSM** *dateDSM, **brathl_refDate** refDate, **brathl_DateJulian** *dateJulian)
- LIBRATHL_API int32_t **brathl_DSM2Seconds** (**brathl_DateDSM** *dateDSM, **brathl_refDate** refDate, **brathl_DateSecond** *dateSeconds)
- LIBRATHL_API int32_t **brathl_DSM2YMDHMSM** (**brathl_DateDSM** *dateDSM, **brathl_DateYMDHMSM** *dateYMDHMSM)

- LIBRATHL_API const char * **brathl_Errno2String** (const int32_t err)
- LIBRATHL_API int32_t **brathl_Julian2DSM** (brathl_DateJulian *dateJulian, brathl_refDate refDate, brathl_DateDSM *dateDSM)
- LIBRATHL_API int32_t **brathl_Julian2Seconds** (brathl_DateJulian *dateJulian, brathl_refDate refDate, brathl_DateSecond *dateSeconds)
- LIBRATHL_API int32_t **brathl_Julian2YMDHMSM** (brathl_DateJulian *dateJulian, brathl_DateYMDHMSM *dateYMDHMSM)
- LIBRATHL_API void **brathl_LoadAliasesDictionary** ()
- LIBRATHL_API int32_t **brathl_NowYMDHMSM** (brathl_DateYMDHMSM *dateYMDHMSM)
- LIBRATHL_API int32_t **brathl_ReadData** (int32_t nbFiles, char **fileNames, const char *recordName, const char *selection, int32_t nbData, char **dataExpressions, char **units, double **results, int32_t sizes[], int32_t *actualSize, int ignoreOutOfRange, int statistics, double defaultValue)
- LIBRATHL_API void **brathl_RegisterAlgorithms** ()
- LIBRATHL_API int32_t **brathl_Seconds2DSM** (brathl_DateSecond *dateSeconds, brathl_refDate refDate, brathl_DateDSM *dateDSM)
- LIBRATHL_API int32_t **brathl_Seconds2Julian** (brathl_DateSecond *dateSeconds, brathl_refDate refDate, brathl_DateJulian *dateJulian)
- LIBRATHL_API int32_t **brathl_Seconds2YMDHMSM** (brathl_DateSecond *dateSeconds, brathl_DateYMDHMSM *dateYMDHMSM)
- LIBRATHL_API int32_t **brathl_YMDHMSM2Cycle** (brathl_mission mission, brathl_DateYMDHMSM *dateYMDHMSM, uint32_t *cycle, uint32_t *pass)
- LIBRATHL_API int32_t **brathl_YMDHMSM2DSM** (brathl_DateYMDHMSM *dateYMDHMSM, brathl_refDate refDate, brathl_DateDSM *dateDSM)
- LIBRATHL_API int32_t **brathl_YMDHMSM2Julian** (brathl_DateYMDHMSM *dateYMDHMSM, brathl_refDate refDate, brathl_DateJulian *dateJulian)
- LIBRATHL_API int32_t **brathl_YMDHMSM2Seconds** (brathl_DateYMDHMSM *dateYMDHMSM, brathl_refDate refDate, brathl_DateSecond *dateSeconds)

Variables

- LIBRATHL_API int **brathl_errno**

7.3.1 Detailed Description

C general interface of BRATHL

7.3.2 Function Documentation

7.3.2.1 LIBRATHL_API const char* brathl_Errno2String (const int32_t err)

Retrieve a string with the error description

With a few exceptions almost all BRATHL functions return an integer that indicate whether the function was able to perform its operations successfully. The return value will be 0 on success and < 0 otherwise. The result is also save in the global variable **brathl_errno** (p. 377) In case you get a negative value.

Parameters

in	err	: error code
----	-----	--------------

Returns

string error description

References BRATHL_ERROR_INVALID_DATE, BRATHL_ERROR_INVALID_DATE_NEGATIVE, BRATHL_ERROR_INVALID_DATE_REF, BRATHL_ERROR_INVALID_DATE_REF_CONV, BRATHL_ERROR_INVALID_DAY,

BRATHL_ERROR_INVALID_DSM, BRATHL_ERROR_INVALID_HOUR, BRATHL_ERROR_INVALID_MINUTE, BRATHL_ERROR_INVALID_MISSION, BRATHL_ERROR_INVALID_MONTH, BRATHL_ERROR_INVALID_MU-SECOND, BRATHL_ERROR_INVALID_NB_PASS, BRATHL_ERROR_INVALID_REPETITION, BRATHL_ERROR_INVALID_SECOND, BRATHL_ERROR_INVALID_YEAR, BRATHL_SUCCESS, BRATHL_WARNING_INVALID_REF_FILE_FIELD, BRATHL_WARNING_INVALID_REF_FILE_FIELDDATE, and BRATHL_WARNING_OPEN_FILE_REF_FILE.

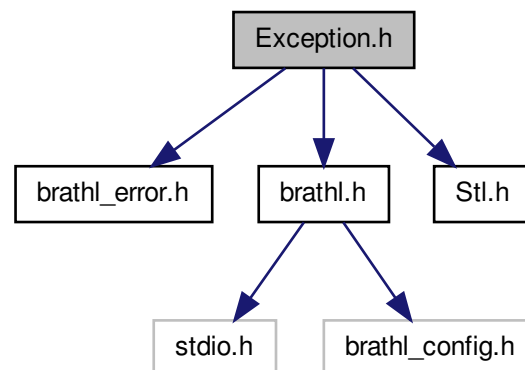
7.3.3 Variable Documentation

7.3.3.1 LIBRATHL_API int brathl_errno

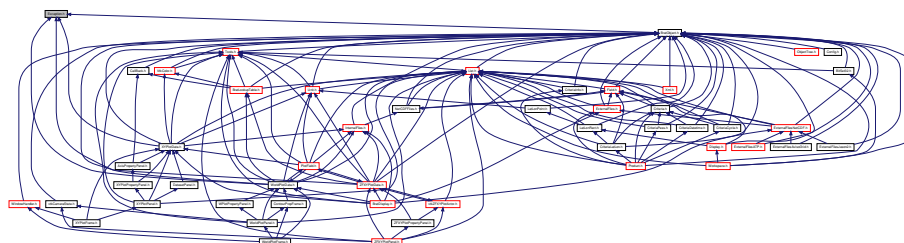
Global variable to save error code

7.4 Exception.h File Reference

```
#include "brathl_error.h"
#include "brathl.h"
#include "Stl.h"
Include dependency graph for Exception.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class **brathl::CAlgorithmException**

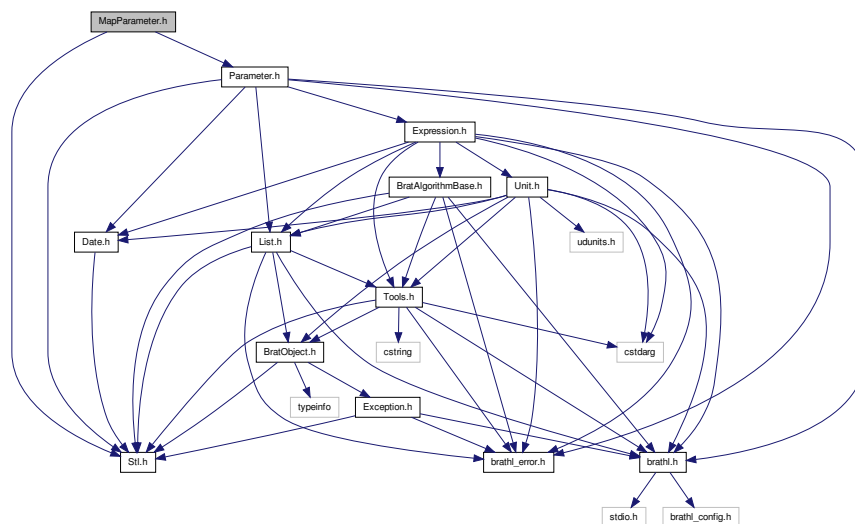
- class **brathl::CException**
- class **brathl::CExpressionException**
- class **brathl::CFileException**
- class **brathl::CLoadAliasesException**
- class **brathl::CMemoryException**
- class **brathl::CParameterException**
- class **brathl::CProductException**
- class **brathl::CUnImplementException**
- class **brathl::CXMLEException**
- class **brathl::CXMLParseException**

7.4.1 Detailed Description

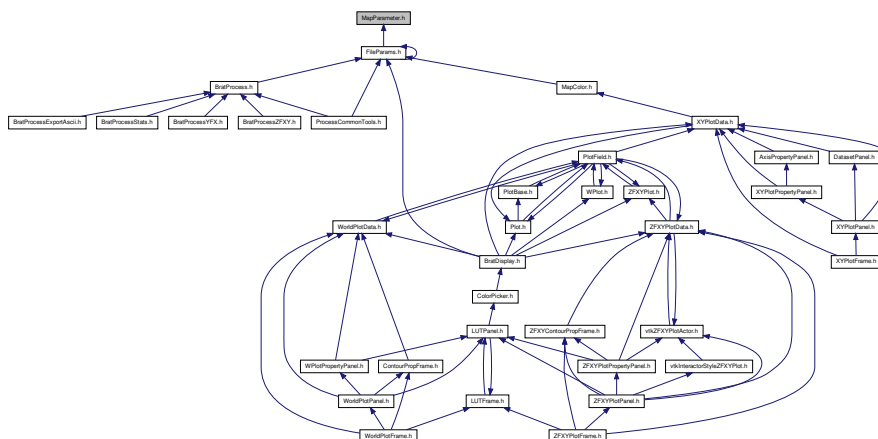
This file contains the various exception classes of brathl

7.5 MapParameter.h File Reference

```
#include "Stl.h"
#include "Parameter.h"
Include dependency graph for MapParameter.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class **brathl::CMapParameter**

Typedefs

- `typedef map< string, CParameter * > brathl::map_parameter`

7.5.1 Detailed Description

Class definition file