

mogasens_csv

Generated by Doxygen 1.8.17

1 Namespace Index	1
1.1 Namespace List	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Namespace Documentation	11
5.1 cl Namespace Reference	11
5.1.1 Typedef Documentation	12
5.1.1.1 column_type	12
5.1.1.2 Expected	13
5.1.2 Enumeration Type Documentation	13
5.1.2.1 Channel	13
5.1.2.2 Column	13
5.1.2.3 CsvFileKind	14
5.1.2.4 Sensor	14
5.1.3 Function Documentation	14
5.1.3.1 CL_SPECIALIZE_COL_TRAITS() [1/11]	14
5.1.3.2 CL_SPECIALIZE_COL_TRAITS() [2/11]	14
5.1.3.3 CL_SPECIALIZE_COL_TRAITS() [3/11]	15
5.1.3.4 CL_SPECIALIZE_COL_TRAITS() [4/11]	15
5.1.3.5 CL_SPECIALIZE_COL_TRAITS() [5/11]	15
5.1.3.6 CL_SPECIALIZE_COL_TRAITS() [6/11]	15
5.1.3.7 CL_SPECIALIZE_COL_TRAITS() [7/11]	15
5.1.3.8 CL_SPECIALIZE_COL_TRAITS() [8/11]	15
5.1.3.9 CL_SPECIALIZE_COL_TRAITS() [9/11]	16
5.1.3.10 CL_SPECIALIZE_COL_TRAITS() [10/11]	16
5.1.3.11 CL_SPECIALIZE_COL_TRAITS() [11/11]	16
5.1.3.12 dataSetAccessor()	16
5.1.3.13 dos2unix()	16
5.1.3.14 isAccelerometer()	17
5.1.3.15 isGyroscope()	18
5.1.3.16 operator<<() [1/4]	18
5.1.3.17 operator<<() [2/4]	18
5.1.3.18 operator<<() [3/4]	19
5.1.3.19 operator<<() [4/4]	20
5.1.3.20 readCsvFile()	20
5.1.3.21 s2n()	21

5.1.3.22 threshold()	21
5.1.3.23 to_string()	22
5.1.3.24 useUnbufferedIo()	22
5.1.4 Variable Documentation	22
5.1.4.1 accelerometerThreshold	23
5.1.4.2 channelCount	23
5.1.4.3 channels	23
5.1.4.4 column_index	23
5.1.4.5 data_set_accessor_v	23
5.1.4.6 gyroscopeThreshold	24
5.1.4.7 sensors	24
5.2 cl::fs Namespace Reference	24
5.2.1 Enumeration Type Documentation	25
5.2.1.1 DirectoryListingOption	25
5.2.2 Function Documentation	25
5.2.2.1 directoryListing()	25
5.2.2.2 formatError()	26
5.2.2.3 operator<()	27
5.2.2.4 operator<<()	27
5.2.2.5 operator==()	28
5.2.2.6 utf16ToUtf8()	28
5.2.2.7 utf8ToUtf16()	29
5.3 cm Namespace Reference	30
5.3.1 Enumeration Type Documentation	30
5.3.1.1 DataSetIdentifier	30
5.4 cs Namespace Reference	31
5.4.1 Enumeration Type Documentation	32
5.4.1.1 FilterKind	32
5.4.1.2 SegmentationKind	33
5.4.2 Function Documentation	33
5.4.2.1 CS_SPECIALIZE_DATA_SET_INFO() [1/20]	33
5.4.2.2 CS_SPECIALIZE_DATA_SET_INFO() [2/20]	33
5.4.2.3 CS_SPECIALIZE_DATA_SET_INFO() [3/20]	34
5.4.2.4 CS_SPECIALIZE_DATA_SET_INFO() [4/20]	34
5.4.2.5 CS_SPECIALIZE_DATA_SET_INFO() [5/20]	34
5.4.2.6 CS_SPECIALIZE_DATA_SET_INFO() [6/20]	34
5.4.2.7 CS_SPECIALIZE_DATA_SET_INFO() [7/20]	34
5.4.2.8 CS_SPECIALIZE_DATA_SET_INFO() [8/20]	34
5.4.2.9 CS_SPECIALIZE_DATA_SET_INFO() [9/20]	35
5.4.2.10 CS_SPECIALIZE_DATA_SET_INFO() [10/20]	35
5.4.2.11 CS_SPECIALIZE_DATA_SET_INFO() [11/20]	35
5.4.2.12 CS_SPECIALIZE_DATA_SET_INFO() [12/20]	35

5.4.2.13 CS_SPECIALIZE_DATA_SET_INFO() [13/20]	35
5.4.2.14 CS_SPECIALIZE_DATA_SET_INFO() [14/20]	35
5.4.2.15 CS_SPECIALIZE_DATA_SET_INFO() [15/20]	36
5.4.2.16 CS_SPECIALIZE_DATA_SET_INFO() [16/20]	36
5.4.2.17 CS_SPECIALIZE_DATA_SET_INFO() [17/20]	36
5.4.2.18 CS_SPECIALIZE_DATA_SET_INFO() [18/20]	36
5.4.2.19 CS_SPECIALIZE_DATA_SET_INFO() [19/20]	36
5.4.2.20 CS_SPECIALIZE_DATA_SET_INFO() [20/20]	36
5.4.2.21 logFiles()	37
5.4.2.22 operator"!="()	38
5.4.2.23 operator<<() [1/3]	39
5.4.2.24 operator<<() [2/3]	39
5.4.2.25 operator<<() [3/3]	40
5.4.2.26 operator==()	40
5.4.2.27 PL_DEFINE_EXCEPTION_TYPE()	40
5.4.2.28 repetitionCount()	41
5.4.3 Variable Documentation	42
5.4.3.1 logPath	42
5.4.3.2 oldLogPath	42
5.5 ctg Namespace Reference	43
5.5.1 Function Documentation	43
5.5.1.1 aboveThreshold()	43
5.5.1.2 averageComparisonValueCalculator()	44
5.5.1.3 halfMaximumComparisonValueCalculator()	45
5.5.1.4 isRelevant()	45
5.5.1.5 percentageOf()	46
5.5.1.6 runAboveThreshold()	47
5.6 fmc Namespace Reference	47
5.6.1 Function Documentation	48
5.6.1.1 adjustHardwareTimestamp()	48
5.6.1.2 convertToUnixLineEndings()	48
5.6.1.3 createBackupFile()	49
5.6.1.4 deleteNonBoschSensors()	50
5.6.1.5 deleteOutOfBoundsValues()	50
5.6.1.6 removeZerosFromField()	50
5.6.1.7 restoreFromBackup()	51
5.6.1.8 writeFile()	51
6 Class Documentation	53
6.1 cl::col_traits< Col > Struct Template Reference	53
6.1.1 Detailed Description	53
6.2 cs::CsvLineBuilder Class Reference	53

6.2.1 Detailed Description	54
6.2.2 Member Typedef Documentation	54
6.2.2.1 <code>this_type</code>	54
6.2.3 Constructor & Destructor Documentation	55
6.2.3.1 <code>CsvLineBuilder()</code>	55
6.2.4 Member Function Documentation	55
6.2.4.1 <code>build()</code>	55
6.2.4.2 <code>dataSet()</code>	56
6.2.4.3 <code>deleteLowVariance()</code>	56
6.2.4.4 <code>deleteTooClose()</code>	57
6.2.4.5 <code>filter()</code>	58
6.2.4.6 <code>isOld()</code>	59
6.2.4.7 <code>kind()</code>	60
6.2.4.8 <code>repetitions()</code>	61
6.2.4.9 <code>segmentationPoints()</code>	62
6.2.4.10 <code>sensor()</code>	63
6.2.4.11 <code>skipWindow()</code>	64
6.2.4.12 <code>windowSize()</code>	65
6.3 <code>cl::data_set_accessor< Chan ></code> Struct Template Reference	66
6.3.1 Detailed Description	66
6.4 <code>cs::data_set_info< Tag ></code> Struct Template Reference	67
6.4.1 Detailed Description	67
6.5 <code>cl::DataPoint</code> Class Reference	67
6.5.1 Detailed Description	67
6.5.2 Constructor & Destructor Documentation	68
6.5.2.1 <code>DataPoint()</code>	68
6.5.3 Member Function Documentation	68
6.5.3.1 <code>channel()</code>	68
6.5.3.2 <code>fileName()</code>	69
6.5.3.3 <code>sensor()</code>	69
6.5.3.4 <code>time()</code>	70
6.5.3.5 <code>value()</code>	70
6.5.4 Friends And Related Function Documentation	70
6.5.4.1 <code>operator<<</code>	71
6.6 <code>cl::DataSet</code> Class Reference	71
6.6.1 Detailed Description	72
6.6.2 Member Typedef Documentation	72
6.6.2.1 <code>ChannelAccessor</code>	72
6.6.2.2 <code>size_type</code>	72
6.6.3 Member Function Documentation	72
6.6.3.1 <code>accelerometerAverage()</code>	72
6.6.3.2 <code>accelerometerMaximum()</code>	73

6.6.3.3 accelerometerX()	73
6.6.3.4 accelerometerY()	74
6.6.3.5 accelerometerZ()	74
6.6.3.6 create()	75
6.6.3.7 extractId()	76
6.6.3.8 fileName()	76
6.6.3.9 gyroscopeAverage()	77
6.6.3.10 gyroscopeMaximum()	78
6.6.3.11 gyroscopeX()	78
6.6.3.12 gyroscopeY()	79
6.6.3.13 gyroscopeZ()	79
6.6.3.14 hardwareTimestamp()	80
6.6.3.15 rowCount()	80
6.6.3.16 time()	80
6.6.3.17 trigger()	81
6.7 cl::Error Class Reference	81
6.7.1 Detailed Description	82
6.7.2 Member Enumeration Documentation	82
6.7.2.1 Kind	82
6.7.3 Constructor & Destructor Documentation	82
6.7.3.1 Error()	82
6.7.4 Member Function Documentation	83
6.7.4.1 file()	83
6.7.4.2 function()	83
6.7.4.3 kind()	84
6.7.4.4 line()	84
6.7.4.5 message()	84
6.7.4.6 raise()	85
6.7.4.7 to_string()	85
6.7.5 Friends And Related Function Documentation	85
6.7.5.1 operator<<	85
6.8 cl::Exception Class Reference	85
6.8.1 Detailed Description	86
6.8.2 Member Typedef Documentation	86
6.8.2.1 base_type	86
6.8.3 Constructor & Destructor Documentation	87
6.8.3.1 Exception() [1/2]	87
6.8.3.2 Exception() [2/2]	87
6.8.4 Member Function Documentation	87
6.8.4.1 file()	87
6.8.4.2 function()	87
6.8.4.3 line()	88

6.9 cl::fs::File Class Reference	88
6.9.1 Detailed Description	88
6.9.2 Constructor & Destructor Documentation	88
6.9.2.1 File()	88
6.9.3 Member Function Documentation	89
6.9.3.1 copyTo()	89
6.9.3.2 create()	90
6.9.3.3 exists()	91
6.9.3.4 moveTo()	91
6.9.3.5 path()	92
6.9.3.6 remove()	93
6.9.3.7 size()	93
6.10 cl::fs::FileStream Class Reference	94
6.10.1 Detailed Description	95
6.10.2 Member Typedef Documentation	95
6.10.2.1 this_type	95
6.10.3 Member Enumeration Documentation	95
6.10.3.1 OpenMode	95
6.10.4 Constructor & Destructor Documentation	96
6.10.4.1 FileStream()	96
6.10.4.2 ~FileStream()	96
6.10.5 Member Function Documentation	96
6.10.5.1 create()	96
6.10.5.2 operator=()	97
6.10.5.3 PL_NONCOPYABLE()	98
6.10.5.4 readAll()	98
6.10.5.5 write()	98
6.11 cs::LogInfo Class Reference	99
6.11.1 Detailed Description	100
6.11.2 Constructor & Destructor Documentation	100
6.11.2.1 LogInfo()	100
6.11.3 Member Function Documentation	100
6.11.3.1 create()	100
6.11.3.2 deleteLowVariance()	101
6.11.3.3 deleteTooClose()	101
6.11.3.4 filterKind()	102
6.11.3.5 isInitialized()	102
6.11.3.6 logFilePath()	102
6.11.3.7 segmentationKind()	103
6.11.3.8 sensor()	103
6.11.3.9 skipWindow()	103
6.11.3.10 windowSize()	104

6.11.4 Friends And Related Function Documentation	104
6.11.4.1 operator"!="	104
6.11.4.2 operator<<	104
6.11.4.3 operator==	105
6.11.5 Member Data Documentation	105
6.11.5.1 invalidSensor	105
6.12 cs::LogLine Class Reference	105
6.12.1 Detailed Description	106
6.12.2 Member Function Documentation	106
6.12.2.1 fileName()	106
6.12.2.2 filePath()	107
6.12.2.3 parse()	107
6.12.2.4 segmentationPointCount()	108
6.12.2.5 sensor()	108
6.12.3 Member Data Documentation	109
6.12.3.1 invalidSensor	109
6.13 cm::ManualSegmentationPoint Class Reference	109
6.13.1 Detailed Description	109
6.13.2 Constructor & Destructor Documentation	110
6.13.2.1 ManualSegmentationPoint()	110
6.13.3 Member Function Documentation	110
6.13.3.1 asMilliseconds()	110
6.13.3.2 frame()	111
6.13.3.3 hour()	111
6.13.3.4 minute()	112
6.13.3.5 readCsvFile()	112
6.13.3.6 second()	112
6.14 cl::fs::Path Class Reference	113
6.14.1 Detailed Description	113
6.14.2 Constructor & Destructor Documentation	113
6.14.2.1 Path()	113
6.14.3 Member Function Documentation	114
6.14.3.1 exists()	114
6.14.3.2 isDirectory()	114
6.14.3.3 isFile()	115
6.14.3.4 str()	116
6.14.4 Friends And Related Function Documentation	117
6.14.4.1 operator<	117
6.14.4.2 operator<<	117
6.14.4.3 operator==	118

7.1 compare_segmentation/CMakeLists.txt File Reference	119
7.1.1 Function Documentation	119
7.1.1.1 set()	119
7.2 compare_segmentation/test/CMakeLists.txt File Reference	119
7.2.1 Function Documentation	119
7.2.1.1 include()	120
7.3 counting/CMakeLists.txt File Reference	120
7.3.1 Function Documentation	120
7.3.1.1 set()	120
7.4 counting/test/CMakeLists.txt File Reference	120
7.4.1 Function Documentation	120
7.4.1.1 include()	120
7.5 csv_lib/CMakeLists.txt File Reference	121
7.5.1 Function Documentation	121
7.5.1.1 set()	121
7.6 csv_lib/test/CMakeLists.txt File Reference	121
7.6.1 Function Documentation	121
7.6.1.1 include()	121
7.7 fix_csv/CMakeLists.txt File Reference	122
7.7.1 Function Documentation	122
7.7.1.1 set()	122
7.8 fix_csv/test/CMakeLists.txt File Reference	122
7.8.1 Function Documentation	122
7.8.1.1 include()	122
7.9 confusion_matrix/CMakeLists.txt File Reference	122
7.9.1 Function Documentation	123
7.9.1.1 set()	123
7.10 confusion_matrix/test/CMakeLists.txt File Reference	123
7.10.1 Function Documentation	123
7.10.1.1 include()	123
7.11 compare_segmentation/include/csv_line.hpp File Reference	123
7.12 compare_segmentation/include/data_set_info.hpp File Reference	124
7.12.1 Macro Definition Documentation	126
7.12.1.1 CS_SPECIALIZE_DATA_SET_INFO	126
7.13 compare_segmentation/include/filter_kind.hpp File Reference	126
7.14 compare_segmentation/include/log_files.hpp File Reference	127
7.15 compare_segmentation/include/log_info.hpp File Reference	128
7.16 compare_segmentation/include/log_line.hpp File Reference	129
7.17 compare_segmentation/include/paths.hpp File Reference	129
7.18 compare_segmentation/include/segmentation_kind.hpp File Reference	130
7.19 compare_segmentation/src/csv_line.cpp File Reference	132
7.20 compare_segmentation/src/data_set_info.cpp File Reference	132

7.21 compare_segmentation/src/filter_kind.cpp File Reference	133
7.22 compare_segmentation/src/log_files.cpp File Reference	133
7.23 compare_segmentation/src/log_info.cpp File Reference	134
7.24 compare_segmentation/src/log_line.cpp File Reference	135
7.25 compare_segmentation/src/main.cpp File Reference	135
7.25.1 Function Documentation	136
7.25.1.1 main()	136
7.26 compare_segmentation/test/main.cpp File Reference	137
7.26.1 Function Documentation	138
7.26.1.1 main()	138
7.27 counting/src/main.cpp File Reference	138
7.27.1 Function Documentation	139
7.27.1.1 main()	139
7.28 counting/test/main.cpp File Reference	140
7.28.1 Function Documentation	141
7.28.1.1 main()	141
7.29 csv_lib/test/main.cpp File Reference	141
7.29.1 Function Documentation	142
7.29.1.1 main()	142
7.30 fix_csv/src/main.cpp File Reference	142
7.30.1 Function Documentation	143
7.30.1.1 main()	143
7.31 fix_csv/test/main.cpp File Reference	143
7.31.1 Function Documentation	144
7.31.1.1 main()	144
7.32 confusion_matrix/src/main.cpp File Reference	144
7.32.1 Function Documentation	144
7.32.1.1 main()	145
7.33 confusion_matrix/test/main.cpp File Reference	145
7.33.1 Function Documentation	145
7.33.1.1 main()	146
7.34 compare_segmentation/src/segmentation_kind.cpp File Reference	146
7.35 compare_segmentation/test/csv_line_test.cpp File Reference	147
7.35.1 Function Documentation	147
7.35.1.1 TEST()	147
7.36 compare_segmentation/test/data_set_info_test.cpp File Reference	147
7.36.1 Function Documentation	148
7.36.1.1 TEST()	148
7.37 compare_segmentation/test/log_files_test.cpp File Reference	149
7.37.1 Function Documentation	149
7.37.1.1 TEST() [1/3]	149
7.37.1.2 TEST() [2/3]	150

7.37.1.3 TEST() [3/3]	150
7.38 compare_segmentation/test/log_info_test.cpp File Reference	150
7.38.1 Function Documentation	151
7.38.1.1 TEST() [1/19]	151
7.38.1.2 TEST() [2/19]	152
7.38.1.3 TEST() [3/19]	152
7.38.1.4 TEST() [4/19]	153
7.38.1.5 TEST() [5/19]	153
7.38.1.6 TEST() [6/19]	154
7.38.1.7 TEST() [7/19]	154
7.38.1.8 TEST() [8/19]	155
7.38.1.9 TEST() [9/19]	155
7.38.1.10 TEST() [10/19]	156
7.38.1.11 TEST() [11/19]	156
7.38.1.12 TEST() [12/19]	157
7.38.1.13 TEST() [13/19]	157
7.38.1.14 TEST() [14/19]	158
7.38.1.15 TEST() [15/19]	158
7.38.1.16 TEST() [16/19]	159
7.38.1.17 TEST() [17/19]	159
7.38.1.18 TEST() [18/19]	160
7.38.1.19 TEST() [19/19]	160
7.39 compare_segmentation/test/log_line_test.cpp File Reference	160
7.39.1 Function Documentation	161
7.39.1.1 TEST() [1/4]	161
7.39.1.2 TEST() [2/4]	161
7.39.1.3 TEST() [3/4]	162
7.39.1.4 TEST() [4/4]	162
7.40 confusion_matrix/include/data_set_identifier.hpp File Reference	163
7.41 confusion_matrix/include/manual_segmentation_point.hpp File Reference	163
7.42 confusion_matrix/src/manual_segmentation_point.cpp File Reference	164
7.42.1 Macro Definition Documentation	165
7.42.1.1 DSI	165
7.43 confusion_matrix/test/manual_segmentation_point_test.cpp File Reference	165
7.43.1 Function Documentation	166
7.43.1.1 TEST() [1/9]	166
7.43.1.2 TEST() [2/9]	166
7.43.1.3 TEST() [3/9]	166
7.43.1.4 TEST() [4/9]	167
7.43.1.5 TEST() [5/9]	167
7.43.1.6 TEST() [6/9]	167
7.43.1.7 TEST() [7/9]	167

7.43.1.8 TEST() [8/9]	167
7.43.1.9 TEST() [9/9]	168
7.44 counting/include/above_threshold.hpp File Reference	168
7.45 counting/include/average_comparison_value_calculator.hpp File Reference	169
7.46 counting/include/half_maximum_comparison_value_calculator.hpp File Reference	170
7.47 counting/include/is_relevant.hpp File Reference	171
7.48 counting/include/percentage_of.hpp File Reference	172
7.49 counting/include/run_above_threshold.hpp File Reference	173
7.50 counting/src/above_threshold.cpp File Reference	173
7.50.1 Macro Definition Documentation	174
7.50.1.1 CL_CHANNEL_X	174
7.50.2 Variable Documentation	174
7.50.2.1 channel	175
7.50.2.2 channelAccessor	175
7.51 counting/src/average_comparison_value_calculator.cpp File Reference	175
7.52 counting/src/half_maximum_comparison_value_calculator.cpp File Reference	176
7.53 counting/src/run_above_threshold.cpp File Reference	176
7.54 counting/test/above_threshold_test.cpp File Reference	177
7.54.1 Macro Definition Documentation	177
7.54.1.1 EXPECT_LONG_DOUBLE_EQ	178
7.54.2 Function Documentation	178
7.54.2.1 TEST()	178
7.55 counting/test/percentage_of_test.cpp File Reference	179
7.55.1 Macro Definition Documentation	179
7.55.1.1 EXPECT_LONG_DOUBLE_EQ	179
7.55.2 Function Documentation	179
7.55.2.1 TEST()	180
7.56 csv_lib/include/cl/channel.hpp File Reference	180
7.56.1 Macro Definition Documentation	181
7.56.1.1 CL_CHANNEL	182
7.56.1.2 CL_CHANNEL_X [1/4]	182
7.56.1.3 CL_CHANNEL_X [2/4]	182
7.56.1.4 CL_CHANNEL_X [3/4]	182
7.56.1.5 CL_CHANNEL_X [4/4]	183
7.57 csv_lib/include/cl/column.hpp File Reference	183
7.57.1 Macro Definition Documentation	184
7.57.1.1 CL_SPECIALIZE_COL_TRAITS	185
7.58 csv_lib/include/cl/data_point.hpp File Reference	185
7.59 csv_lib/include/cl/data_set.hpp File Reference	186
7.60 csv_lib/include/cl/dos2unix.hpp File Reference	187
7.61 csv_lib/include/cl/error.hpp File Reference	188
7.61.1 Macro Definition Documentation	188

7.61.1.1 CL_ERROR_KIND	189
7.61.1.2 CL_ERROR_KIND_X	189
7.61.1.3 CL_UNEXPECTED	189
7.62 csv_lib/include/cl/exception.hpp File Reference	189
7.62.1 Macro Definition Documentation	190
7.62.1.1 CL_THROW	190
7.62.1.2 CL_THROW_FMT	190
7.63 csv_lib/include/cl/fs/directory_listing.hpp File Reference	191
7.64 csv_lib/include/cl/fs/file.hpp File Reference	192
7.65 csv_lib/include/cl/fs/file_stream.hpp File Reference	193
7.66 csv_lib/include/cl/fs/path.hpp File Reference	194
7.67 csv_lib/include/cl/fs/separator.hpp File Reference	194
7.67.1 Macro Definition Documentation	195
7.67.1.1 CL_FS_SEPARATOR	195
7.68 csv_lib/include/cl/fs/windows.hpp File Reference	196
7.68.1 Detailed Description	197
7.69 csv_lib/include/cl/read_csv_file.hpp File Reference	197
7.70 csv_lib/include/cl/s2n.hpp File Reference	198
7.71 csv_lib/include/cl/sensor.hpp File Reference	198
7.71.1 Macro Definition Documentation	200
7.71.1.1 CL_SENSOR	200
7.71.1.2 CL_SENSOR_X [1/2]	200
7.71.1.3 CL_SENSOR_X [2/2]	200
7.72 csv_lib/include/cl/to_string.hpp File Reference	201
7.73 csv_lib/include/cl/use_unbuffered_io.hpp File Reference	202
7.74 csv_lib/src/cl/channel.cpp File Reference	202
7.74.1 Macro Definition Documentation	203
7.74.1.1 CL_CHANNEL_X [1/2]	203
7.74.1.2 CL_CHANNEL_X [2/2]	203
7.75 csv_lib/src/cl/data_point.cpp File Reference	204
7.75.1 Function Documentation	204
7.75.1.1 channel()	204
7.75.1.2 fileName()	205
7.75.1.3 sensor()	205
7.75.1.4 time()	206
7.75.1.5 value()	206
7.76 csv_lib/src/cl/data_set.cpp File Reference	207
7.77 csv_lib/src/cl/dos2unix.cpp File Reference	207
7.78 csv_lib/src/cl/error.cpp File Reference	208
7.78.1 Macro Definition Documentation	208
7.78.1.1 CL_ERROR_KIND_X	209
7.79 csv_lib/src/cl/exception.cpp File Reference	209

7.80 csv_lib/src/cl/fs/directory_listing.cpp File Reference	209
7.81 csv_lib/src/cl/fs/file.cpp File Reference	210
7.82 csv_lib/src/cl/fs/file_stream.cpp File Reference	210
7.83 csv_lib/src/cl/fs/path.cpp File Reference	211
7.84 csv_lib/src/cl/fs/windows.cpp File Reference	212
7.85 csv_lib/src/cl/read_csv_file.cpp File Reference	212
7.86 csv_lib/src/cl/sensor.cpp File Reference	213
7.86.1 Macro Definition Documentation	214
7.86.1.1 CL_SENSOR_X	214
7.87 csv_lib/src/cl/use_unbuffered_io.cpp File Reference	214
7.88 csv_lib/test/channel_test.cpp File Reference	215
7.88.1 Function Documentation	215
7.88.1.1 TEST() [1/4]	215
7.88.1.2 TEST() [2/4]	215
7.88.1.3 TEST() [3/4]	216
7.88.1.4 TEST() [4/4]	216
7.89 csv_lib/test/column_test.cpp File Reference	217
7.89.1 Function Documentation	217
7.89.1.1 TEST() [1/2]	217
7.89.1.2 TEST() [2/2]	217
7.90 csv_lib/test/data_point_test.cpp File Reference	218
7.90.1 Function Documentation	218
7.90.1.1 TEST() [1/2]	218
7.90.1.2 TEST() [2/2]	219
7.90.2 Variable Documentation	219
7.90.2.1 dp	219
7.91 csv_lib/test/data_set_test.cpp File Reference	220
7.91.1 Macro Definition Documentation	220
7.91.1.1 EXPECT_LONG_DOUBLE_EQ	220
7.91.2 Function Documentation	220
7.91.2.1 TEST() [1/4]	221
7.91.2.2 TEST() [2/4]	221
7.91.2.3 TEST() [3/4]	222
7.91.2.4 TEST() [4/4]	223
7.92 csv_lib/test/directory_listing_test.cpp File Reference	224
7.92.1 Function Documentation	225
7.92.1.1 TEST() [1/3]	225
7.92.1.2 TEST() [2/3]	225
7.92.1.3 TEST() [3/3]	226
7.93 csv_lib/test/error_test.cpp File Reference	226
7.93.1 Function Documentation	227
7.93.1.1 TEST() [1/4]	227

7.93.1.2 TEST() [2/4]	227
7.93.1.3 TEST() [3/4]	227
7.93.1.4 TEST() [4/4]	227
7.93.2 Variable Documentation	227
7.93.2.1 error	228
7.94 csv_lib/test/exception_test.cpp File Reference	228
7.94.1 Function Documentation	228
7.94.1.1 TEST()	228
7.95 csv_lib/test/read_csv_file_test.cpp File Reference	229
7.95.1 Function Documentation	229
7.95.1.1 TEST() [1/2]	229
7.95.1.2 TEST() [2/2]	230
7.96 csv_lib/test/s2n_test.cpp File Reference	230
7.96.1 Function Documentation	231
7.96.1.1 TEST() [1/3]	231
7.96.1.2 TEST() [2/3]	231
7.96.1.3 TEST() [3/3]	232
7.97 csv_lib/test/sensor_test.cpp File Reference	232
7.97.1 Function Documentation	233
7.97.1.1 TEST() [1/2]	233
7.97.1.2 TEST() [2/2]	233
7.98 csv_lib/test/to_string_test.cpp File Reference	233
7.98.1 Function Documentation	234
7.98.1.1 TEST()	234
7.99 fix_csv/include/adjust_hardware_timestamp.hpp File Reference	234
7.100 fix_csv/include/convert_to_unix_line_endings.hpp File Reference	235
7.101 fix_csv/include/create_backup_file.hpp File Reference	236
7.102 fix_csv/include/delete_non_bosch_sensors.hpp File Reference	237
7.103 fix_csv/include/delete_out_of_bounds_values.hpp File Reference	238
7.104 fix_csv/include/remove_zeros_from_field.hpp File Reference	239
7.105 fix_csv/include/restore_from_backup.hpp File Reference	240
7.106 fix_csv/include/write_file.hpp File Reference	241
7.107 fix_csv/src/adjust_hardware_timestamp.cpp File Reference	242
7.108 fix_csv/src/convert_to_unix_line_endings.cpp File Reference	243
7.109 fix_csv/src/create_backup_file.cpp File Reference	244
7.110 fix_csv/src/delete_non_bosch_sensors.cpp File Reference	244
7.110.1 Macro Definition Documentation	245
7.110.1.1 CL_SENSOR_X	245
7.111 fix_csv/src/delete_out_of_bounds_values.cpp File Reference	245
7.112 fix_csv/src/remove_zeros_from_field.cpp File Reference	246
7.113 fix_csv/src/restore_from_backup.cpp File Reference	246
7.114 fix_csv/src/write_file.cpp File Reference	247

7.115 fix_csv/test/adjust_hardware_timestamp_test.cpp File Reference	248
7.115.1 Function Documentation	248
7.115.1.1 TEST() [1/5]	248
7.115.1.2 TEST() [2/5]	249
7.115.1.3 TEST() [3/5]	249
7.115.1.4 TEST() [4/5]	250
7.115.1.5 TEST() [5/5]	250
7.116 fix_csv/test/remove_zeros_from_field_test.cpp File Reference	250
7.116.1 Function Documentation	251
7.116.1.1 TEST() [1/6]	251
7.116.1.2 TEST() [2/6]	252
7.116.1.3 TEST() [3/6]	252
7.116.1.4 TEST() [4/6]	253
7.116.1.5 TEST() [5/6]	253
7.116.1.6 TEST() [6/6]	254
Index	255

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

cl	11
cl::fs	24
cm	30
cs	31
ctg	43
fmc	47

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

cl::col_traits< Col >	53
cs::CsvLineBuilder	53
cl::data_set_accessor< Chan >	66
cs::data_set_info< Tag >	67
cl::DataPoint	67
cl::DataSet	71
cl::Error	81
std::exception	
std::runtime_error	
cl::Exception	85
cl::fs::File	88
cl::fs::FileStream	94
cs::LogInfo	99
cs::LogLine	105
cm::ManualSegmentationPoint	109
cl::fs::Path	113

Chapter 3

Class Index

3.1 Class List

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

cl::col_traits< Col >	53
cs::CsvLineBuilder	
Builder for a CSV line	53
cl::data_set_accessor< Chan >	66
cs::data_set_info< Tag >	
Meta function for data set tags	67
cl::DataPoint	67
cl::DataSet	71
cl::Error	81
cl::Exception	85
cl::fs::File	
Represents a file	88
cl::fs::FileStream	
A binary file stream	94
cs::LogInfo	
Information about a log file	99
cs::LogLine	
A line out of a log file	105
cm::ManualSegmentationPoint	
Type used to represent a manual segmentation point	109
cl::fs::Path	
A filesystem path	113

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

compare_segmentation/include/csv_line.hpp	123
compare_segmentation/include/data_set_info.hpp	124
compare_segmentation/include/filter_kind.hpp	126
compare_segmentation/include/log_files.hpp	127
compare_segmentation/include/log_info.hpp	128
compare_segmentation/include/log_line.hpp	129
compare_segmentation/include/paths.hpp	129
compare_segmentation/include/segmentation_kind.hpp	130
compare_segmentation/src/csv_line.cpp	132
compare_segmentation/src/data_set_info.cpp	132
compare_segmentation/src/filter_kind.cpp	133
compare_segmentation/src/log_files.cpp	133
compare_segmentation/src/log_info.cpp	134
compare_segmentation/src/log_line.cpp	135
compare_segmentation/src/main.cpp	135
compare_segmentation/src/segmentation_kind.cpp	146
compare_segmentation/test/csv_line_test.cpp	147
compare_segmentation/test/data_set_info_test.cpp	147
compare_segmentation/test/log_files_test.cpp	149
compare_segmentation/test/log_info_test.cpp	150
compare_segmentation/test/log_line_test.cpp	160
compare_segmentation/test/main.cpp	137
confusion_matrix/include/data_set_identifier.hpp	163
confusion_matrix/include/manual_segmentation_point.hpp	163
confusion_matrix/src/main.cpp	144
confusion_matrix/src/manual_segmentation_point.cpp	164
confusion_matrix/test/main.cpp	145
confusion_matrix/test/manual_segmentation_point_test.cpp	165
counting/include/above_threshold.hpp	168
counting/include/average_comparison_value_calculator.hpp	169
counting/include/half_maximum_comparison_value_calculator.hpp	170
counting/include/is_relevant.hpp	171
counting/include/percentage_of.hpp	172
counting/include/run_above_threshold.hpp	173
counting/src/above_threshold.cpp	173

counting/src/average_comparison_value_calculator.cpp	175
counting/src/half_maximum_comparison_value_calculator.cpp	176
counting/src/main.cpp	138
counting/src/run_above_threshold.cpp	176
counting/test/above_threshold_test.cpp	177
counting/test/main.cpp	140
counting/test/percentage_of_test.cpp	179
csv_lib/include/cl/channel.hpp	180
csv_lib/include/cl/column.hpp	183
csv_lib/include/cl/data_point.hpp	185
csv_lib/include/cl/data_set.hpp	186
csv_lib/include/cl/dos2unix.hpp	187
csv_lib/include/cl/error.hpp	188
csv_lib/include/cl/exception.hpp	189
csv_lib/include/cl/read_csv_file.hpp	197
csv_lib/include/cl/s2n.hpp	198
csv_lib/include/cl/sensor.hpp	198
csv_lib/include/cl/to_string.hpp	201
csv_lib/include/cl/use_unbuffered_io.hpp	202
csv_lib/include/cl/fs/directory_listing.hpp	191
csv_lib/include/cl/fs/file.hpp	192
csv_lib/include/cl/fs/file_stream.hpp	193
csv_lib/include/cl/fs/path.hpp	194
csv_lib/include/cl/fs/sePARATOR.hpp	194
csv_lib/include/cl/fs/windows.hpp	
Contains Microsoft Windows specific functions	196
csv_lib/src/cl/channel.cpp	202
csv_lib/src/cl/data_point.cpp	204
csv_lib/src/cl/data_set.cpp	207
csv_lib/src/cl/dos2unix.cpp	207
csv_lib/src/cl/error.cpp	208
csv_lib/src/cl/exception.cpp	209
csv_lib/src/cl/read_csv_file.cpp	212
csv_lib/src/cl/sensor.cpp	213
csv_lib/src/cl/use_unbuffered_io.cpp	214
csv_lib/src/cl/fs/directory_listing.cpp	209
csv_lib/src/cl/fs/file.hpp	210
csv_lib/src/cl/fs/file_stream.hpp	210
csv_lib/src/cl/fs/path.hpp	211
csv_lib/src/cl/fs/windows.hpp	212
csv_lib/test/channel_test.cpp	215
csv_lib/test/column_test.cpp	217
csv_lib/test/data_point_test.cpp	218
csv_lib/test/data_set_test.cpp	220
csv_lib/test/directory_listing_test.cpp	224
csv_lib/test/error_test.cpp	226
csv_lib/test/exception_test.cpp	228
csv_lib/test/main.cpp	141
csv_lib/test/read_csv_file_test.cpp	229
csv_lib/test/s2n_test.cpp	230
csv_lib/test/sensor_test.cpp	232
csv_lib/test/to_string_test.cpp	233
fix_csv/include/adjust_hardware_timestamp.hpp	234
fix_csv/include/convert_to_unix_line_endings.hpp	235
fix_csv/include/create_backup_file.hpp	236
fix_csv/include/delete_non_bosch_sensors.hpp	237
fix_csv/include/delete_out_of_bounds_values.hpp	238
fix_csv/include/remove_zeros_from_field.hpp	239

fix_csv/include/restore_from_backup.hpp	240
fix_csv/include/write_file.hpp	241
fix_csv/src/adjust_hardware_timestamp.cpp	242
fix_csv/src/convert_to_unix_line_endings.cpp	243
fix_csv/src/create_backup_file.cpp	244
fix_csv/src/delete_non_bosch_sensors.cpp	244
fix_csv/src/delete_out_of_bounds_values.cpp	245
fix_csv/src/main.cpp	142
fix_csv/src/remove_zeros_from_field.cpp	246
fix_csv/src/restore_from_backup.cpp	246
fix_csv/src/write_file.cpp	247
fix_csv/test/adjust_hardware_timestamp_test.cpp	248
fix_csv/test/main.cpp	143
fix_csv/test/remove_zeros_from_field_test.cpp	250

Chapter 5

Namespace Documentation

5.1 cl Namespace Reference

Namespaces

- `fs`

Classes

- struct `col_traits`
- struct `data_set_accessor`
- class `DataPoint`
- class `DataSet`
- class `Error`
- class `Exception`

Typedefs

- template<Column Col>
using `column_type` = typename `col_traits`< Col >::type
- template<typename Ty >
using `Expected` = tl::expected< Ty, `Error` >

Enumerations

- enum `Channel` : std::uint64_t { `Channel::CL_CHANNEL_X`, `Channel::CL_CHANNEL` }
- enum `Column` : std::size_t {
`Column::Time`, `Column::HardwareTimestamp`, `Column::ExtractId`, `Column::Trigger`,
`Column::AccelerometerX`, `Column::AccelerometerY`, `Column::AccelerometerZ`, `Column::GyroscopeX`,
`Column::GyroscopeY`, `Column::GyroscopeZ`, `Column::SamplingRate` }
- enum `CsvFileKind` { `CsvFileKind::Raw`, `CsvFileKind::Fixed` }
- enum `Sensor` : std::uint64_t { `Sensor::CL_SENSOR_X`, `Sensor::CL_SENSOR` }

Functions

- `DataSet::ChannelAccessor dataSetAccessor (Channel channel)`
- `std::ostream & operator<< (std::ostream &os, Channel channel)`
- `bool isAccelerometer (Channel channel)`
- `bool isGyroscope (Channel channel)`
- `long double threshold (Channel channel)`
- `CL_SPECIALIZE_COL_TRAITS (Column::Time, long double)`
- `CL_SPECIALIZE_COL_TRAITS (Column::HardwareTimestamp, std::uint64_t)`
- `CL_SPECIALIZE_COL_TRAITS (Column::ExtractId, Sensor)`
- `CL_SPECIALIZE_COL_TRAITS (Column::Trigger, std::uint64_t)`
- `CL_SPECIALIZE_COL_TRAITS (Column::AccelerometerX, long double)`
- `CL_SPECIALIZE_COL_TRAITS (Column::AccelerometerY, long double)`
- `CL_SPECIALIZE_COL_TRAITS (Column::AccelerometerZ, long double)`
- `CL_SPECIALIZE_COL_TRAITS (Column::GyroscopeX, long double)`
- `CL_SPECIALIZE_COL_TRAITS (Column::GyroscopeY, long double)`
- `CL_SPECIALIZE_COL_TRAITS (Column::GyroscopeZ, long double)`
- `CL_SPECIALIZE_COL_TRAITS (Column::SamplingRate, std::uint64_t)`
- `std::vector< pl::byte > dos2unix (const void *p, std::size_t size)`

Converts DOS / Microsoft Windows line endings to UNIX line endings.
- `Expected< std::vector< std::vector< std::string > > > readCsvFile (pl::string_view csvFilePath, std::vector< std::string > *columnNames=nullptr, CsvFileKind csvFileKind=CsvFileKind::Fixed) noexcept`
- `template<typename Integer> Expected< Integer > s2n (const std::string &str, std::size_t *pos=nullptr, [[maybe_unused]] int base=10)`
- `std::ostream & operator<< (std::ostream &os, Sensor sensor)`
- `template<typename Ty> std::string to_string (const Ty &ty)`
- `void useUnbufferedIo ()`
- `std::ostream & operator<< (std::ostream &os, const DataPoint &dataPoint)`
- `std::ostream & operator<< (std::ostream &os, const Error &error)`

Variables

- `constexpr std::size_t channelCount`
- `constexpr std::array< Channel, channelCount > channels`
- `template<Channel Chan> constexpr CL_CHANNEL DataSet::ChannelAccessor data_set_accessor_v = data_set_accessor<Chan>::f`
- `constexpr long double accelerometerThreshold {1.99L}`
- `constexpr long double gyroscopeThreshold {1999.99L}`
- `template<Column Col> constexpr std::size_t column_index = col_traits<Col>::index`
- `constexpr std::array< Sensor, 4 > sensors`

5.1.1 Typedef Documentation

5.1.1.1 column_type

```
template<Column Col>
using cl::column_type = typedef typename col_traits<Col>::type
```

Definition at line 49 of file column.hpp.

5.1.1.2 Expected

```
template<typename Ty >
using cl::Expected = typedef tl::expected<Ty, Error>
```

Definition at line 63 of file error.hpp.

5.1.2 Enumeration Type Documentation

5.1.2.1 Channel

```
enum cl::Channel : std::uint64_t [strong]
```

Enumerator

CL_CHANNEL←_X	
CL_CHANNEL	

Definition at line 20 of file channel.hpp.

5.1.2.2 Column

```
enum cl::Column : std::size_t [strong]
```

Enumerator

Time	
HardwareTimestamp	
ExtractId	
Trigger	
AccelerometerX	
AccelerometerY	
AccelerometerZ	
GyroscopeX	
GyroscopeY	
GyroscopeZ	
SamplingRate	

Definition at line 9 of file column.hpp.

5.1.2.3 CsvFileKind

```
enum cl::CsvFileKind [strong]
```

Enumerator

Raw	
Fixed	

Definition at line 11 of file read_csv_file.hpp.

5.1.2.4 Sensor

```
enum cl::Sensor : std::uint64_t [strong]
```

Enumerator

CL_SENSOR_X	
CL_SENSOR	

Definition at line 15 of file sensor.hpp.

5.1.3 Function Documentation

5.1.3.1 CL_SPECIALIZE_COL_TRAITS() [1/11]

```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::AccelerometerX ,
    long double )
```

5.1.3.2 CL_SPECIALIZE_COL_TRAITS() [2/11]

```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::AccelerometerY ,
    long double )
```

5.1.3.3 CL_SPECIALIZE_COL_TRAITS() [3/11]

```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::AccelerometerZ ,
    long double   )
```

5.1.3.4 CL_SPECIALIZE_COL_TRAITS() [4/11]

```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::ExtractId ,
    Sensor   )
```

5.1.3.5 CL_SPECIALIZE_COL_TRAITS() [5/11]

```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::GyroscopeX ,
    long double   )
```

5.1.3.6 CL_SPECIALIZE_COL_TRAITS() [6/11]

```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::GyroscopeY ,
    long double   )
```

5.1.3.7 CL_SPECIALIZE_COL_TRAITS() [7/11]

```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::GyroscopeZ ,
    long double   )
```

5.1.3.8 CL_SPECIALIZE_COL_TRAITS() [8/11]

```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::HardwareTimestamp ,
    std::uint64_t   )
```

5.1.3.9 CL_SPECIALIZE_COL_TRAITS() [9/11]

```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::SamplingRate ,
    std::uint64_t )
```

5.1.3.10 CL_SPECIALIZE_COL_TRAITS() [10/11]

```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::Time ,
    long double )
```

5.1.3.11 CL_SPECIALIZE_COL_TRAITS() [11/11]

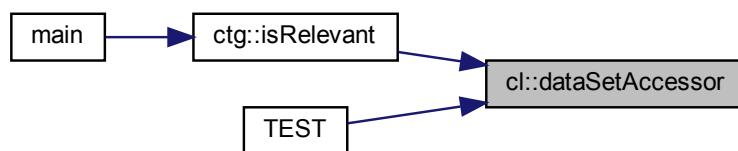
```
cl::CL_SPECIALIZE_COL_TRAITS (
    Column::Trigger ,
    std::uint64_t )
```

5.1.3.12 dataSetAccessor()

```
DataSet::ChannelAccessor cl::dataSetAccessor (
    Channel channel )
```

Definition at line 15 of file channel.cpp.

Here is the caller graph for this function:



5.1.3.13 dos2unix()

```
std::vector< pl::byte > cl::dos2unix (
    const void * p,
    std::size_t size )
```

Converts DOS / Microsoft Windows line endings to UNIX line endings.

Parameters

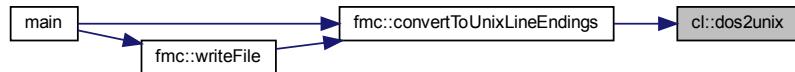
<i>p</i>	The beginning of the data to convert.
<i>size</i>	The size of the data to convert in bytes.

Returns

The resulting byte array.

Definition at line 4 of file dos2unix.cpp.

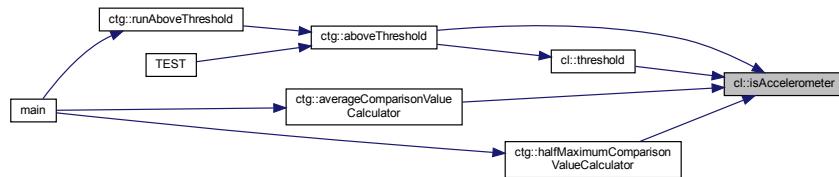
Here is the caller graph for this function:

**5.1.3.14 isAccelerometer()**

```
bool cl::isAccelerometer (
    Channel channel )
```

Definition at line 45 of file channel.cpp.

Here is the caller graph for this function:

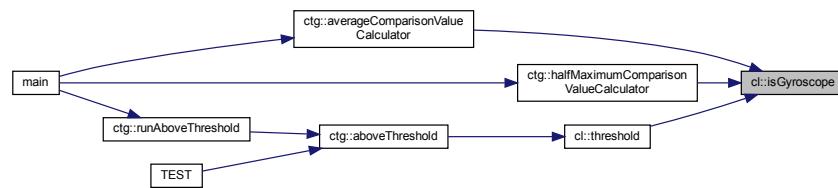


5.1.3.15 `isGyroscope()`

```
bool cl::isGyroscope (
    Channel channel )
```

Definition at line 50 of file channel.cpp.

Here is the caller graph for this function:



5.1.3.16 `operator<<()` [1/4]

```
std::ostream & cl::operator<< (
    std::ostream & os,
    Channel channel )
```

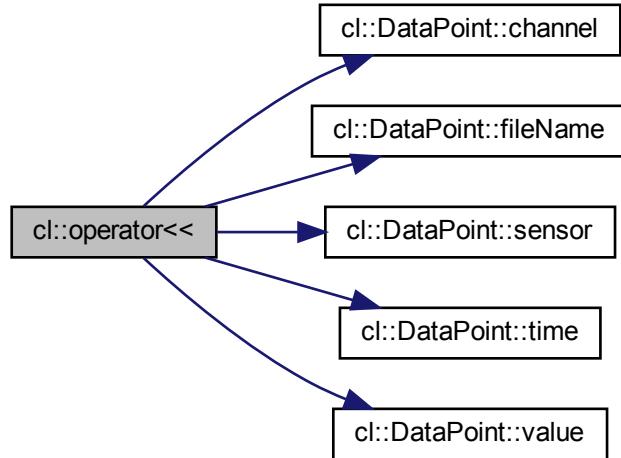
Definition at line 32 of file channel.cpp.

5.1.3.17 `operator<<()` [2/4]

```
std::ostream& cl::operator<< (
    std::ostream & os,
    const DataPoint & dataPoint )
```

Definition at line 10 of file data_point.cpp.

Here is the call graph for this function:

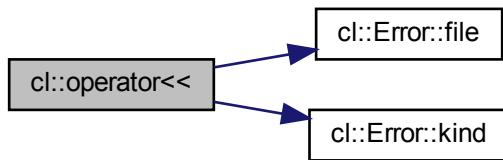


5.1.3.18 `operator<<()` [3/4]

```
std::ostream& cl::operator<< (
    std::ostream & os,
    const Error & error )
```

Definition at line 30 of file error.cpp.

Here is the call graph for this function:



5.1.3.19 operator<<() [4/4]

```
std::ostream & cl::operator<< (
    std::ostream & os,
    Sensor sensor )
```

Definition at line 8 of file sensor.cpp.

Here is the call graph for this function:

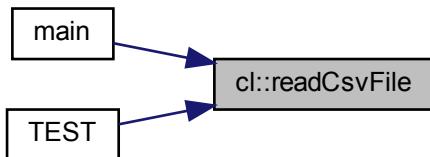


5.1.3.20 readCsvFile()

```
Expected< std::vector< std::vector< std::string > > > cl::readCsvFile (
    pl::string_view csvFilePath,
    std::vector< std::string > * columnNames = nullptr,
    CsvFileKind csvFileKind = CsvFileKind::Fixed ) [noexcept]
```

Definition at line 50 of file read_csv_file.cpp.

Here is the caller graph for this function:



5.1.3.21 s2n()

```
template<typename Integer >
Expected<Integer> cl::s2n (
    const std::string & str,
    std::size_t * pos = nullptr,
    [[maybe_unused]] int base = 10 ) [inline]
```

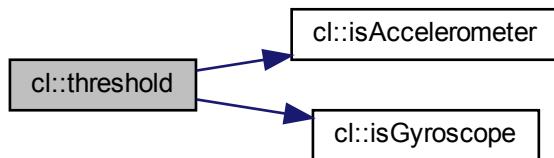
Definition at line 16 of file s2n.hpp.

5.1.3.22 threshold()

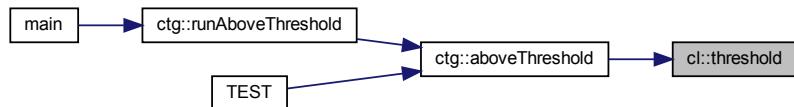
```
long double cl::threshold (
    Channel channel )
```

Definition at line 55 of file channel.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

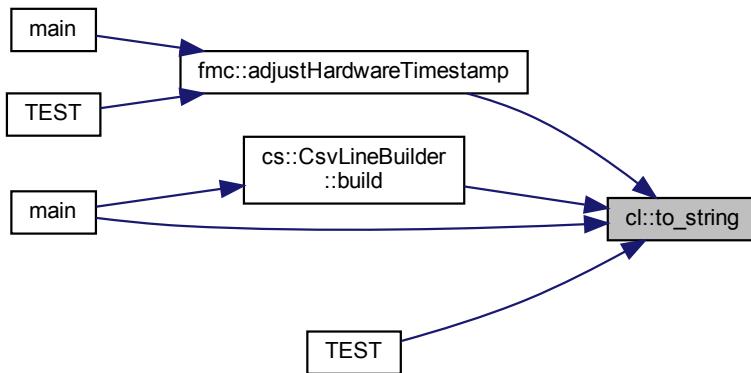


5.1.3.23 `to_string()`

```
template<typename Ty >
std::string cl::to_string (
    const Ty & ty ) [inline]
```

Definition at line 16 of file `to_string.hpp`.

Here is the caller graph for this function:



5.1.3.24 `useUnbufferedIo()`

```
void cl::useUnbufferedIo ( )
```

Definition at line 9 of file `use_unbuffered_io.cpp`.

Here is the caller graph for this function:



5.1.4 Variable Documentation

5.1.4.1 accelerometerThreshold

```
constexpr long double cl::accelerometerThreshold {1.99L} [inline], [constexpr]
```

Definition at line 61 of file channel.hpp.

5.1.4.2 channelCount

```
constexpr std::size_t cl::channelCount [inline], [constexpr]
```

Initial value:

```
{0  
#define CL_CHANNEL_X(enumerator, value, dataSetAccessor)  
    CL_CHANNEL  
}
```

Definition at line 26 of file channel.hpp.

5.1.4.3 channels

```
constexpr std::array<Channel, channelCount> cl::channels [inline], [constexpr]
```

Initial value:

```
{  
#define CL_CHANNEL_X(enm, v, a)  
    CL_CHANNEL  
} }
```

Definition at line 32 of file channel.hpp.

5.1.4.4 column_index

```
template<Column Col>  
constexpr std::size_t cl::column_index = col_traits<Col>::index [inline], [constexpr]
```

Definition at line 46 of file column.hpp.

5.1.4.5 data_set_accessor_v

```
template<Channel Chan>  
constexpr CL_CHANNEL DataSet::ChannelAccessor cl::data_set_accessor_v = data_set_accessor<Chan>↔  
::f [inline], [constexpr]
```

Definition at line 51 of file channel.hpp.

5.1.4.6 gyroscopeThreshold

```
constexpr long double cl::gyroscopeThreshold {1999.99L} [inline], [constexpr]
```

Definition at line 62 of file channel.hpp.

5.1.4.7 sensors

```
constexpr std::array<Sensor, 4> cl::sensors [inline], [constexpr]
```

Initial value:

```
{}  
#define CL_SENSOR_X(enm, v)  
    CL_SENSOR  
){}
```

Definition at line 21 of file sensor.hpp.

5.2 cl::fs Namespace Reference

Classes

- class [File](#)
Represents a file.
- class [FileStream](#)
A binary file stream.
- class [Path](#)
A filesystem path.

Enumerations

- enum [DirectoryListingOption](#) { [DirectoryListingOption::None](#), [DirectoryListingOption::ExcludeDotAndDotDot](#) }
Options for directoryListing.

Functions

- [Expected< std::vector< Path > > directoryListing](#) (const [Path](#) &directoryPath, [DirectoryListingOption](#) directoryListingOption=[DirectoryListingOption::ExcludeDotAndDotDot](#))
Creates a listing of the contents of a directory.
- [std::wstring utf8ToUtf16](#) ([pl::string_view](#) utf8)
Converts a UTF-8 encoded string to a UTF-16 encoded wstring.
- [std::string utf16ToUtf8](#) ([pl::wstring_view](#) utf16)
Converts a UTF-16 encoded wide character string to UTF-8 string.
- [std::wstring formatError](#) (DWORD errorCode)
Formats a WINAPI error code to a UTF-16 encoded wide character string.
- [std::ostream & operator<<](#) ([std::ostream](#) &os, const [Path](#) &path)
- [bool operator<](#) (const [Path](#) &lhs, const [Path](#) &rhs) noexcept
- [bool operator==](#) (const [Path](#) &lhs, const [Path](#) &rhs) noexcept

5.2.1 Enumeration Type Documentation

5.2.1.1 DirectoryListingOption

```
enum cl::fs::DirectoryListingOption [strong]
```

Options for directoryListing.

Enumerator

None	No option
ExcludeDotAndDotDot	Exclude the . and .. directories

Definition at line 13 of file directory_listing.hpp.

5.2.2 Function Documentation

5.2.2.1 directoryListing()

```
Expected< std::vector< Path > > cl::fs::directoryListing (
    const Path & directoryPath,
    DirectoryListingOption directoryListingOption = DirectoryListingOption::ExcludeDotAndDotDot
)
```

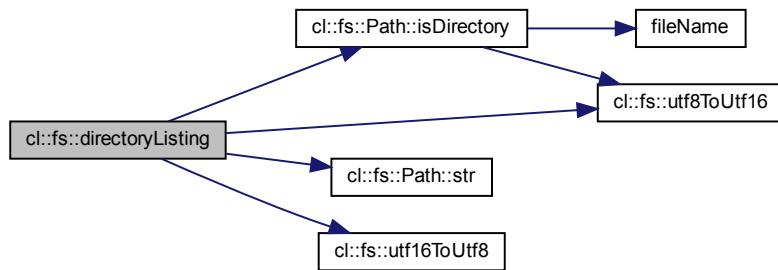
Creates a listing of the contents of a directory.

Parameters

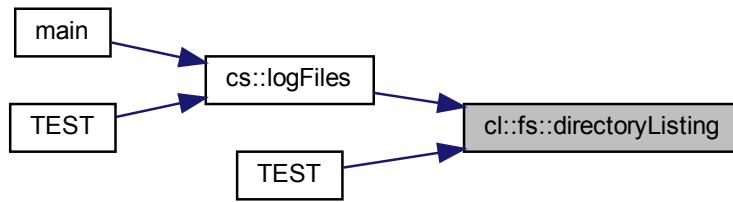
<i>directoryPath</i>	The directory to list.
<i>directoryListingOption</i>	The option to use.

Definition at line 24 of file directory_listing.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



5.2.2.2 formatError()

```
std::wstring cl::fs::formatError (
    DWORD errorCode )
```

Formats a WINAPI error code to a UTF-16 encoded wide character string.

Parameters

<code>errorCode</code>	The WINAPI error code.
------------------------	------------------------

Returns

The resulting UTF-16 encoded wide character string.

Note

Most WINAPIs expect UTF-16 encoded wide character strings, but we don't want to pollute the code base with UTF-16 strings.

Warning

Wide characters are only 16 bit wide on Microsoft Windows, they're 32 bit on GNU / Linux.

Definition at line 89 of file windows.cpp.

5.2.2.3 operator<()

```
bool cl::fs::operator< (
    const Path & lhs,
    const Path & rhs ) [noexcept]
```

Parameters

<i>lhs</i>	The left hand side operand.
<i>rhs</i>	The right hand side operand.

Returns

true if lhs < rhs; otherwise false.

Definition at line 27 of file path.cpp.

5.2.2.4 operator<<()

```
std::ostream& cl::fs::operator<< (
    std::ostream & os,
    const Path & path )
```

Parameters

<i>os</i>	the ostream to print to.
<i>path</i>	The path to print.

Returns

os

Definition at line 22 of file path.cpp.

5.2.2.5 operator==()

```
bool cl::fs::operator== (
    const Path & lhs,
    const Path & rhs ) [noexcept]
```

Parameters

<i>lhs</i>	The left hand side operand.
<i>rhs</i>	The right hand side operand.

Returns

true if *lhs* and *rhs* are equal.

Definition at line 32 of file path.cpp.

5.2.2.6 utf16ToUtf8()

```
std::string cl::fs::utf16ToUtf8 (
    pl::wstring_view utf16 )
```

Converts a UTF-16 encoded wide character string to UTF-8 string.

Parameters

<i>utf16</i>	The UTF-16 encoded wide character string to convert.
--------------	--

Returns

The resulting UTF-8 string.

Note

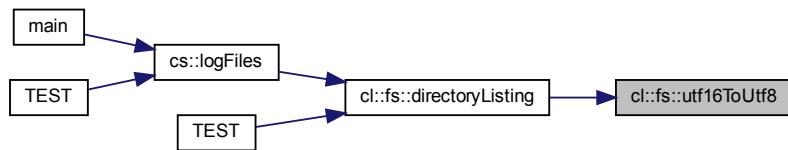
Most WINAPIs expect UTF-16 encoded wide character strings, but we don't want to pollute the code base with UTF-16 strings.

Warning

Wide characters are only 16 bit wide on Microsoft Windows, they're 32 bit on GNU / Linux.

Definition at line 61 of file windows.cpp.

Here is the caller graph for this function:



5.2.2.7 utf8ToUtf16()

```
std::wstring cl::fs::utf8ToUtf16 (
    pl::string_view utf8 )
```

Converts a UTF-8 encoded string to a UTF-16 encoded wstring.

Parameters

<i>utf8</i>	The UTF-8 encoded string to convert.
-------------	--------------------------------------

Returns

The resulting UTF-16 string.

Note

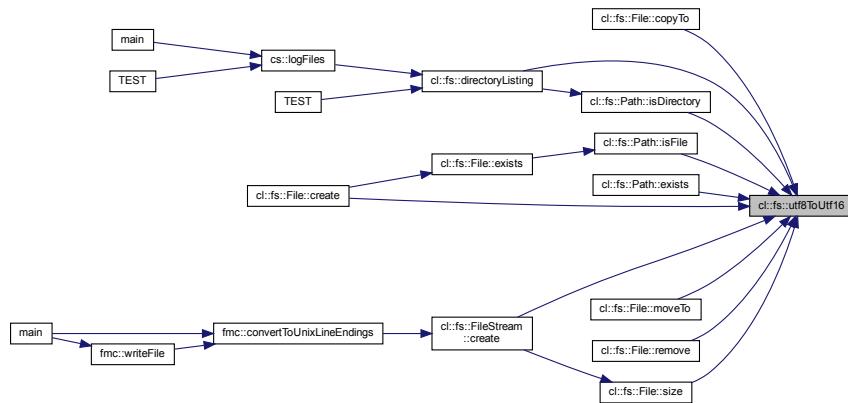
Most WINAPIs expect UTF-16 encoded wide character strings, but we don't want to pollute the code base with UTF-16 strings.

Warning

Wide characters are only 16 bit wide on Microsoft Windows, they're 32 bit on GNU / Linux.

Definition at line 35 of file windows.cpp.

Here is the caller graph for this function:



5.3 cm Namespace Reference

Classes

- class [ManualSegmentationPoint](#)

Type used to represent a manual segmentation point.

Enumerations

- enum [DataSetIdentifier](#) {
 `DataSetIdentifier::Felix_11_17_39`, `DataSetIdentifier::Felix_12_50_00`, `DataSetIdentifier::Felix_13_00_09`,
`DataSetIdentifier::Mike_14_07_33`, `DataSetIdentifier::Mike_14_14_32`, `DataSetIdentifier::Mike_14_20_28`, `DataSetIdentifier::Marsi_14_59_59`,
`DataSetIdentifier::Marsi_15_13_22`, `DataSetIdentifier::Marsi_15_31_36`, `DataSetIdentifier::Jan_1`, `DataSetIdentifier::Jan_2`, `DataSetIdentifier::Jan_3`,
`DataSetIdentifier::Andre_1`, `DataSetIdentifier::Andre_2`, `DataSetIdentifier::Andre_3`, `DataSetIdentifier::Andre_Squats_1`,
`DataSetIdentifier::Andre_Squats_2`, `DataSetIdentifier::Lucas_1`, `DataSetIdentifier::Lucas_2`, `DataSetIdentifier::Lucas_3`
}

5.3.1 Enumeration Type Documentation

5.3.1.1 DataSetIdentifier

```
enum cm::DataSetIdentifier [strong]
```

Enumerator

<code>Felix_11_17_39</code>	
<code>Felix_12_50_00</code>	

Enumerator

Felix_13_00_09	
Mike_14_07_33	
Mike_14_14_32	
Mike_14_20_28	
Marsi_14_59_59	
Marsi_15_13_22	
Marsi_15_31_36	
Jan_1	
Jan_2	
Jan_3	
Andre_1	
Andre_2	
Andre_3	
Andre_Squats	← _1
Andre_Squats	← _2
Lucas_1	
Lucas_2	
Lucas_3	

Definition at line 5 of file `data_set_identifier.hpp`.

5.4 cs Namespace Reference

Classes

- class [CsvLineBuilder](#)
Builder for a CSV line.
- struct [data_set_info](#)
Meta function for data set tags.
- class [LogInfo](#)
Information about a log file.
- class [LogLine](#)
A line out of a log file.

Enumerations

- enum [FilterKind](#) { `FilterKind::Butterworth`, `FilterKind::MovingAverage` }
Type for the different kinds of filters.
- enum [SegmentationKind](#) : pl::byte { `SegmentationKind::Minima` = 0b0000'0001, `SegmentationKind::Maxima` = 0b0000'0010, `SegmentationKind::Both` = `Minima` | `Maxima` }
The segmentation kind.

Functions

- `PL_DEFINE_EXCEPTION_TYPE` (`NoSuchDataSetException`, `std::logic_error`)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Felix1`, "11.17.39", 24)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Felix2`, "12.50.00", 20)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Felix3`, "13.00.09", 15)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Marcelle1`, "14.59.59", 10)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Marcelle2`, "15.13.22", 16)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Marcelle3`, "15.31.36", 18)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Mike1`, "14.07.33", 26)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Mike2`, "14.14.32", 22)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Mike3`, "14.20.28", 18)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Andre1`, "Andre_liegestuetzen1", 27)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Andre2`, "Andre_liegestuetzen2", 20)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Andre3`, "Andre_liegestuetzen3", 17)
- `CS_SPECIALIZE_DATA_SET_INFO` (`AndreSquats1`, "Andre_Squats", 30)
- `CS_SPECIALIZE_DATA_SET_INFO` (`AndreSquats2`, "Andre_Squats2", 49)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Jan1`, "Jan_liegestuetzen1", 25)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Jan2`, "Jan_liegestuetzen2", 19)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Jan3`, "Jan_liegestuetzen3", 13)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Lucas1`, "Lucas_liegestuetzen1", 24)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Lucas2`, "Lucas_liegestuetzen2", 19)
- `CS_SPECIALIZE_DATA_SET_INFO` (`Lucas3`, "Lucas_liegestuetzen3", 11)
- `std::uint64_t repetitionCount (pl::string_view dataSet)`

Fetches the repetition count for a given data set identified by its string.
- `std::ostream & operator<< (std::ostream &os, FilterKind filterKind)`

Prints a FilterKind to an ostream.
- `cl::Expected< std::vector< cl::fs::Path > > logFiles (pl::string_view directoryPath)`

Fetches the paths to the log files in the given directory.
- `std::ostream & operator<< (std::ostream &os, SegmentationKind segmentationKind)`

Prints a SegmentationKind to an ostream.
- `bool operator== (const LogInfo &lhs, const LogInfo &rhs) noexcept`
- `bool operator!= (const LogInfo &lhs, const LogInfo &rhs) noexcept`
- `std::ostream & operator<< (std::ostream &os, const LogInfo &logInfo)`

Variables

- `constexpr pl::string_view logPath {"segmentation_comparison/logs"}`

Relative path to the directory containing the preprocessed log files.
- `constexpr pl::string_view oldLogPath {"segmentation_comparison/logs/old"}`

Relative path to the directory containing the old log files.

5.4.1 Enumeration Type Documentation

5.4.1.1 FilterKind

```
enum cs::FilterKind [strong]
```

Type for the different kinds of filters.

Enumerator

Butterworth	
MovingAverage	

Definition at line 9 of file filter_kind.hpp.

5.4.1.2 SegmentationKind

```
enum cs::SegmentationKind : pl::byte [strong]
```

The segmentation kind.

Enumerator

Minima	Segmentation by local minima
Maxima	Segmentation by local maxima
Both	Segmentation by both local extrema

Definition at line 12 of file segmentation_kind.hpp.

5.4.2 Function Documentation

5.4.2.1 CS_SPECIALIZE_DATA_SET_INFO() [1/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Andre1 ,
    "Andre_liegestuetzen1" ,
    27 )
```

5.4.2.2 CS_SPECIALIZE_DATA_SET_INFO() [2/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Andre2 ,
    "Andre_liegestuetzen2" ,
    20 )
```

5.4.2.3 CS_SPECIALIZE_DATA_SET_INFO() [3/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Andre3 ,
    "Andre_liegestuetzen3" ,
    17 )
```

5.4.2.4 CS_SPECIALIZE_DATA_SET_INFO() [4/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    AndreSquats1 ,
    "Andre_Squats" ,
    30 )
```

5.4.2.5 CS_SPECIALIZE_DATA_SET_INFO() [5/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    AndreSquats2 ,
    "Andre_Squats2" ,
    49 )
```

5.4.2.6 CS_SPECIALIZE_DATA_SET_INFO() [6/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Felix1 ,
    "11.17.39" ,
    24 )
```

5.4.2.7 CS_SPECIALIZE_DATA_SET_INFO() [7/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Felix2 ,
    "12.50.00" ,
    20 )
```

5.4.2.8 CS_SPECIALIZE_DATA_SET_INFO() [8/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Felix3 ,
    "13.00.09" ,
    15 )
```

5.4.2.9 CS_SPECIALIZE_DATA_SET_INFO() [9/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Jan1 ,
    "Jan_liegestuetzen1" ,
    25  )
```

5.4.2.10 CS_SPECIALIZE_DATA_SET_INFO() [10/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Jan2 ,
    "Jan_liegestuetzen2" ,
    19  )
```

5.4.2.11 CS_SPECIALIZE_DATA_SET_INFO() [11/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Jan3 ,
    "Jan_liegestuetzen3" ,
    13  )
```

5.4.2.12 CS_SPECIALIZE_DATA_SET_INFO() [12/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Lucas1 ,
    "Lucas_liegestuetzen1" ,
    24  )
```

5.4.2.13 CS_SPECIALIZE_DATA_SET_INFO() [13/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Lucas2 ,
    "Lucas_liegestuetzen2" ,
    19  )
```

5.4.2.14 CS_SPECIALIZE_DATA_SET_INFO() [14/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Lucas3 ,
    "Lucas_liegestuetzen3" ,
    11  )
```

5.4.2.15 CS_SPECIALIZE_DATA_SET_INFO() [15/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Marcelle1 ,
    "14.59.59" ,
    10   )
```

5.4.2.16 CS_SPECIALIZE_DATA_SET_INFO() [16/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Marcelle2 ,
    "15.13.22" ,
    16   )
```

5.4.2.17 CS_SPECIALIZE_DATA_SET_INFO() [17/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Marcelle3 ,
    "15.31.36" ,
    18   )
```

5.4.2.18 CS_SPECIALIZE_DATA_SET_INFO() [18/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Mike1 ,
    "14.07.33" ,
    26   )
```

5.4.2.19 CS_SPECIALIZE_DATA_SET_INFO() [19/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Mike2 ,
    "14.14.32" ,
    22   )
```

5.4.2.20 CS_SPECIALIZE_DATA_SET_INFO() [20/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
    Mike3 ,
    "14.20.28" ,
    18   )
```

5.4.2.21 logFiles()

```
cl::Expected< std::vector< cl::fs::Path > > cs::logFiles (
```

```
    pl::string_view directoryPath )
```

Fetches the paths to the log files in the given directory.

Parameters

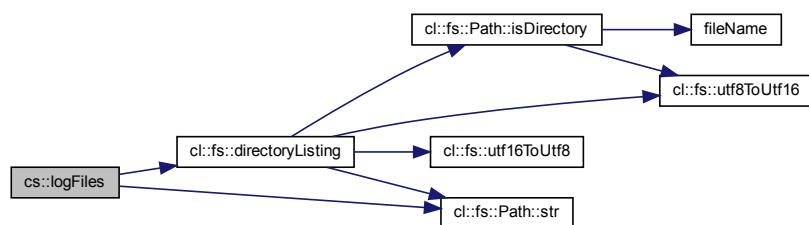
<i>directoryPath</i>	The path to a directory to search for log files.
----------------------	--

Returns

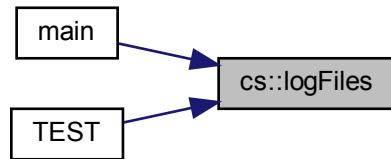
The log files found or an error.

Definition at line 9 of file log_files.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

**5.4.2.22 operator"!=()**

```
bool cs::operator!= (
    const LogInfo & lhs,
    const LogInfo & rhs ) [noexcept]
```

Parameters

<i>lhs</i>	The left hand side operand.
<i>rhs</i>	The right hand side operand.

Returns

true if lhs and rhs are considered not equal; otherwise false.

Definition at line 287 of file log_info.cpp.

5.4.2.23 operator<<() [1/3]

```
std::ostream& cs::operator<< (
    std::ostream & os,
    const LogInfo & logInfo )
```

Parameters

<i>os</i>	The ostream to print to.
<i>logInfo</i>	The LogInfo to print.

Returns

os

Definition at line 292 of file log_info.cpp.

5.4.2.24 operator<<() [2/3]

```
std::ostream & cs::operator<< (
    std::ostream & os,
    FilterKind filterKind )
```

Prints a FilterKind to an ostream.

Parameters

<i>os</i>	The ostream to print to.
<i>filterKind</i>	The FilterKind to print.

Returns

os

Definition at line 6 of file filter_kind.cpp.

5.4.2.25 operator<<() [3/3]

```
std::ostream & cs::operator<< (
    std::ostream & os,
    SegmentationKind segmentationKind )
```

Prints a SegmentationKind to an ostream.

Parameters

<i>os</i>	The ostream to print to.
<i>segmentationKind</i>	The SegmentationKind to print.

Returns

os

Definition at line 6 of file segmentation_kind.cpp.

5.4.2.26 operator==()

```
bool cs::operator== (
    const LogInfo & lhs,
    const LogInfo & rhs ) [noexcept]
```

Parameters

<i>lhs</i>	The left hand side operand.
<i>rhs</i>	The right hand side operand.

Returns

true if *lhs* and *rhs* are considered equal; otherwise false.

Definition at line 264 of file log_info.cpp.

5.4.2.27 PL_DEFINE_EXCEPTION_TYPE()

```
cs::PL_DEFINE_EXCEPTION_TYPE (
    NoSuchDataSetException ,
    std::logic_error )
```

5.4.2.28 repetitionCount()

```
std::uint64_t cs::repetitionCount (
    pl::string_view dataSet )
```

Fetches the repetition count for a given data set identified by its string.

Parameters

<code>dataSet</code>	The data set to fetch the repetition count of.
----------------------	--

Returns

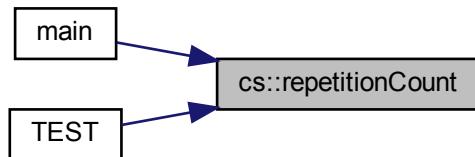
The repetition count of `dataSet`.

Warning

`dataSet` may not be invalid!

Definition at line 10 of file `data_set_info.cpp`.

Here is the caller graph for this function:



5.4.3 Variable Documentation

5.4.3.1 logPath

```
constexpr pl::string_view cs::logPath {"segmentation_comparison/logs"} [inline], [constexpr]
```

Relative path to the directory containing the preprocessed log files.

Definition at line 9 of file `paths.hpp`.

5.4.3.2 oldLogPath

```
constexpr pl::string_view cs::oldLogPath {"segmentation_comparison/logs/old"} [inline], [constexpr]
```

Relative path to the directory containing the old log files.

Definition at line 14 of file `paths.hpp`.

5.5 ctg Namespace Reference

Functions

- `std::vector< cl::DataPoint > aboveThreshold (const cl::DataSet &dataSet, long double accelerometerThreshold, long double gyroscopeThreshold)`
- `long double averageComparisonValueCalculator (cl::Sensor sensor, cl::Channel channel, const cl::DataSet &dataSet)`
- `long double halfMaximumComparisonValueCalculator (cl::Sensor sensor, cl::Channel channel, const cl::DataSet &dataSet)`
- `template<typename ComparisonValueCalculator> bool isRelevant (cl::Sensor sensor, cl::Channel channel, const cl::DataSet &dataSet, ComparisonValueCalculator comparisonValueCalculator)`
- `constexpr long double percentageOf (std::size_t amount, std::size_t totalCount) noexcept`
- `void runAboveThreshold (std::ostream &aboveThresholdLogFileStream, const cl::DataSet &dataSet)`

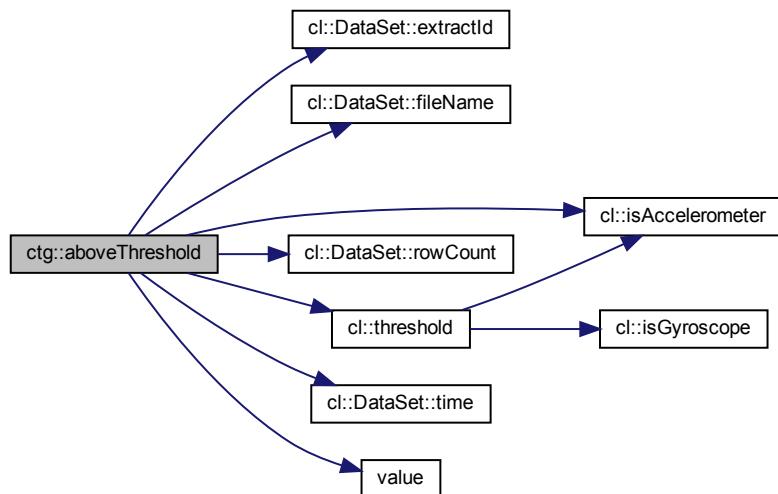
5.5.1 Function Documentation

5.5.1.1 aboveThreshold()

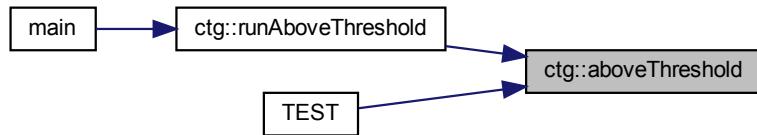
```
std::vector< cl::DataPoint > ctg::aboveThreshold (
    const cl::DataSet & dataSet,
    long double accelerometerThreshold,
    long double gyroscopeThreshold )
```

Definition at line 28 of file above_threshold.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



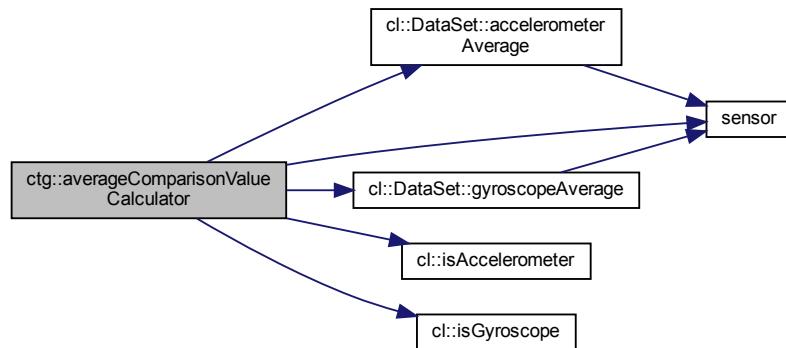
5.5.1.2 averageComparisonValueCalculator()

```

long double ctg::averageComparisonValueCalculator (
    cl::Sensor sensor,
    cl::Channel channel,
    const cl::DataSet & dataSet )
  
```

Definition at line 10 of file average_comparison_value_calculator.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

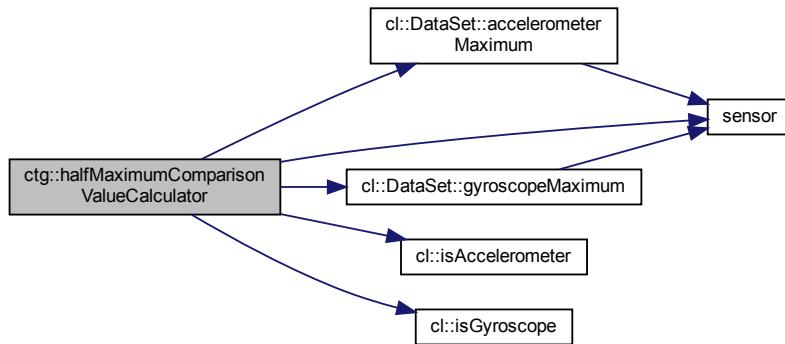


5.5.1.3 halfMaximumComparisonValueCalculator()

```
long double ctg::halfMaximumComparisonValueCalculator (
    cl::Sensor sensor,
    cl::Channel channel,
    const cl::DataSet & dataSet )
```

Definition at line 10 of file half_maximum_comparison_value_calculator.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

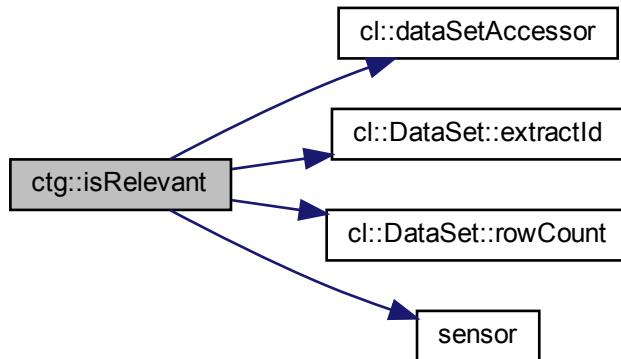


5.5.1.4 isRelevant()

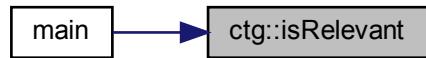
```
template<typename ComparisonValueCalculator >
bool ctg::isRelevant (
    cl::Sensor sensor,
    cl::Channel channel,
    const cl::DataSet & dataSet,
    ComparisonValueCalculator comparisonValueCalculator )
```

Definition at line 11 of file is_relevant.hpp.

Here is the call graph for this function:



Here is the caller graph for this function:



5.5.1.5 `percentageOf()`

```
constexpr long double ctg::percentageOf (
    std::size_t amount,
    std::size_t totalCount ) [constexpr], [noexcept]
```

Definition at line 6 of file `percentage_of.hpp`.

Here is the caller graph for this function:

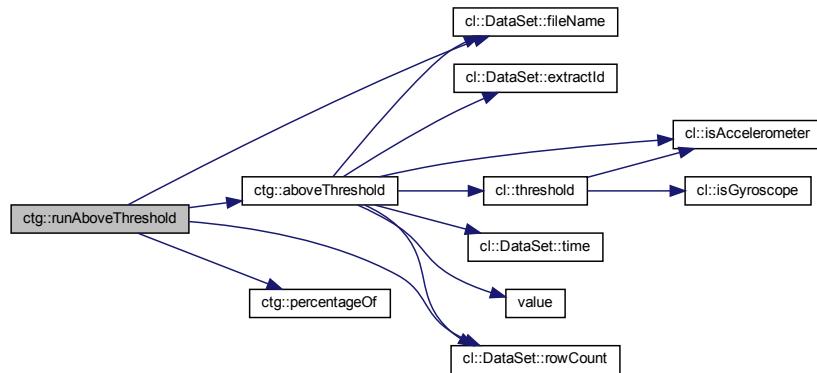


5.5.1.6 runAboveThreshold()

```
void ctg::runAboveThreshold (
    std::ostream & aboveThresholdLogFileStream,
    const cl::DataSet & dataSet )
```

Definition at line 14 of file run_above_threshold.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



5.6 fmc Namespace Reference

Functions

- void [adjustHardwareTimestamp](#) (std::string *cellContent, const std::string &nextRowHardwareTimestamp, std::uint64_t *overflowCount)
- bool [convertToUnixLineEndings](#) (const std::string &csvPath)
- bool [createBackupFile](#) (const std::string &csvFilePath, const std::string &backupFilePath)
- void [deleteNonBoschSensors](#) (std::vector< std::vector< std::string >> *data)
- [cl::Expected< void >](#) [deleteOutOfBoundsValues](#) (std::vector< std::vector< std::string >> *data)
- void [removeZerosFromField](#) (std::string *field)
- bool [restoreFromBackup](#) (const std::string &csvFilePath, const std::string &backupFilePath)
- bool [writeFile](#) (pl::string_view csvPath, pl::string_view csvFileExtension, const std::vector< std::string > &columnNames, const std::vector< std::vector< std::string >> &data)

5.6.1 Function Documentation

5.6.1.1 `adjustHardwareTimestamp()`

```
void fmc::adjustHardwareTimestamp (
    std::string * cellContent,
    const std::string & nextRowHardwareTimestamp,
    std::uint64_t * overflowCount )
```

Definition at line 16 of file `adjust_hardware_timestamp.cpp`.

Here is the call graph for this function:



Here is the caller graph for this function:

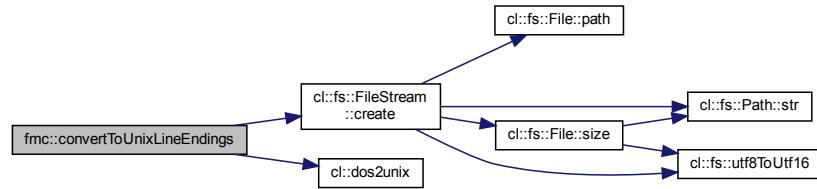


5.6.1.2 `convertToUnixLineEndings()`

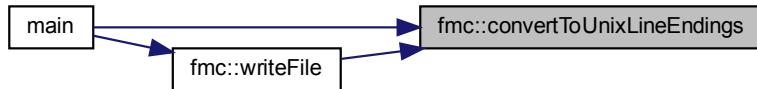
```
bool fmc::convertToUnixLineEndings (
    const std::string & csvPath )
```

Definition at line 18 of file `convert_to_unix_line_endings.cpp`.

Here is the call graph for this function:



Here is the caller graph for this function:



5.6.1.3 `createBackupFile()`

```
bool fmc::createBackupFile (
    const std::string & csvFilePath,
    const std::string & backupFilePath )
```

Definition at line 6 of file `create_backup_file.cpp`.

Here is the caller graph for this function:



5.6.1.4 **deleteNonBoschSensors()**

```
void fmc::deleteNonBoschSensors (
    std::vector< std::vector< std::string >> * data )
```

Definition at line 30 of file `delete_non_bosch_sensors.cpp`.

Here is the caller graph for this function:



5.6.1.5 **deleteOutOfBoundsValues()**

```
cl::Expected< void > fmc::deleteOutOfBoundsValues (
    std::vector< std::vector< std::string >> * data )
```

Definition at line 29 of file `delete_out_of_bounds_values.cpp`.

Here is the caller graph for this function:

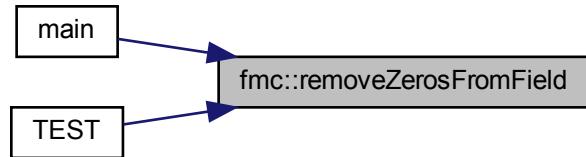


5.6.1.6 **removeZerosFromField()**

```
void fmc::removeZerosFromField (
    std::string * field )
```

Definition at line 6 of file `remove_zeros_from_field.cpp`.

Here is the caller graph for this function:



5.6.1.7 restoreFromBackup()

```
bool fmc::restoreFromBackup (
    const std::string & csvFilePath,
    const std::string & backupFilePath )
```

Definition at line 11 of file restore_from_backup.cpp.

Here is the caller graph for this function:

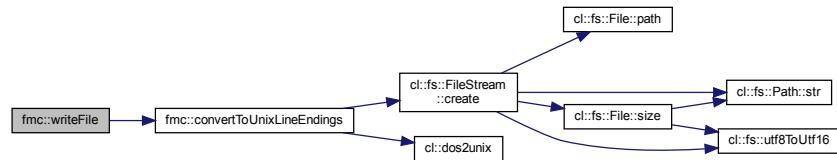


5.6.1.8 writeFile()

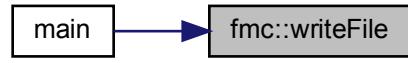
```
bool fmc::writeFile (
    pl::string_view csvPath,
    pl::string_view csvFileExtension,
    const std::vector< std::string > & columnNames,
    const std::vector< std::vector< std::string >> & data )
```

Definition at line 12 of file write_file.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



Chapter 6

Class Documentation

6.1 cl::col_traits< Col > Struct Template Reference

```
#include <column.hpp>
```

6.1.1 Detailed Description

```
template<Column Col>
struct cl::col_traits< Col >
```

Definition at line 24 of file column.hpp.

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

- csv_lib/include/cl/column.hpp

6.2 cs::CsvLineBuilder Class Reference

Builder for a CSV line.

```
#include <csv_line.hpp>
```

Public Types

- using [this_type = CsvLineBuilder](#)

Public Member Functions

- [CsvLineBuilder \(\)](#)
Creates an empty, invalid CsvLineBuilder.
- [this_type & skipWindow \(bool value\)](#)
Write accessor for the skip window property.
- [this_type & deleteTooClose \(bool value\)](#)
Write accessor for the delete too close property.
- [this_type & deleteLowVariance \(bool value\)](#)
Write accessor for the delete low variance property.
- [this_type & kind \(SegmentationKind value\)](#)
Write accessor for the kind property.
- [this_type & windowSize \(std::uint64_t value\)](#)
Write accessor for the window size property.
- [this_type & filter \(FilterKind value\)](#)
Write accessor for the filter property.
- [this_type & dataSet \(std::string value\)](#)
Write accessor for the data set property.
- [this_type & sensor \(std::uint64_t value\)](#)
Write accessor for the sensor property.
- [this_type & repetitions \(std::uint64_t value\)](#)
Write accessor for the repetitions property.
- [this_type & segmentationPoints \(std::uint64_t value\)](#)
Write accessor for the segmentation points property.
- [this_type & isOld \(bool value\)](#)
Write accessor for the is old property.
- [std::vector< std::string > build \(\) const](#)
Builds the CSV line as a vector containing the cells of the CSV line.

6.2.1 Detailed Description

Builder for a CSV line.

Builder type for a CSV line. All write accessors have to be called before the build member function is called!

Definition at line 21 of file csv_line.hpp.

6.2.2 Member Typedef Documentation

6.2.2.1 this_type

```
using cs::CsvLineBuilder::this_type = CsvLineBuilder
```

Definition at line 23 of file csv_line.hpp.

6.2.3 Constructor & Destructor Documentation

6.2.3.1 CsvLineBuilder()

```
cs::CsvLineBuilder::CsvLineBuilder ( )
```

Creates an empty, invalid [CsvLineBuilder](#).

Definition at line 44 of file csv_line.cpp.

6.2.4 Member Function Documentation

6.2.4.1 build()

```
std::vector< std::string > cs::CsvLineBuilder::build ( ) const
```

Builds the CSV line as a vector containing the cells of the CSV line.

Returns

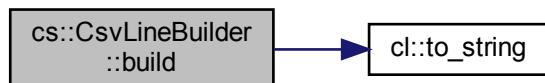
The resulting vector of strings.

Warning

May only be called after all the write accessors have been called.

Definition at line 124 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.2 dataSet()

```
CsvLineBuilder & cs::CsvLineBuilder::dataSet (
    std::string value )
```

Write accessor for the data set property.

Parameters

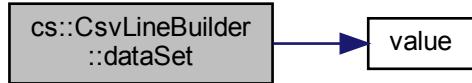
value	The value to use.
-------	-------------------

Returns

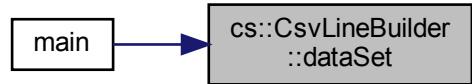
*this

Definition at line 94 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.3 deleteLowVariance()

```
CsvLineBuilder & cs::CsvLineBuilder::deleteLowVariance (
    bool value )
```

Write accessor for the delete low variance property.

Parameters

<code>value</code>	The value to use.
--------------------	-------------------

Returns`*this`

Definition at line 70 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.4 deleteTooClose()

```
CsvLineBuilder & cs::CsvLineBuilder::deleteTooClose (\n    bool value )
```

Write accessor for the delete too close property.

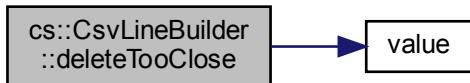
Parameters

<code>value</code>	The value to use.
--------------------	-------------------

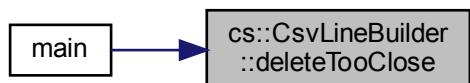
Returns`*this`

Definition at line 64 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.5 filter()

```
CsvLineBuilder & cs::CsvLineBuilder::filter (
    FilterKind value )
```

Write accessor for the filter property.

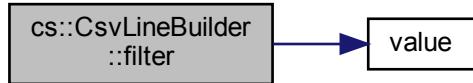
Parameters

<code>value</code>	The value to use.
--------------------	-------------------

Returns`*this`

Definition at line 88 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.6 isOld()

```
CsvLineBuilder & cs::CsvLineBuilder::isOld ( bool value )
```

Write accessor for the is old property.

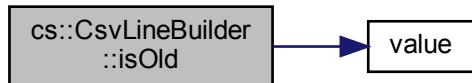
Parameters

<code>value</code>	The value to use.
--------------------	-------------------

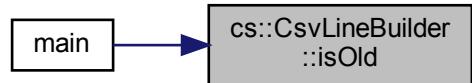
Returns`*this`

Definition at line 118 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.7 kind()

```
CsvLineBuilder & cs::CsvLineBuilder::kind (
    SegmentationKind value )
```

Write accessor for the kind property.

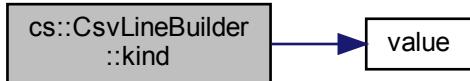
Parameters

<code>value</code>	The value to use.
--------------------	-------------------

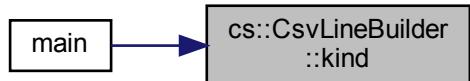
Returns`*this`

Definition at line 76 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.8 repetitions()

```
CsvLineBuilder & cs::CsvLineBuilder::repetitions ( std::uint64_t value )
```

Write accessor for the repetitions property.

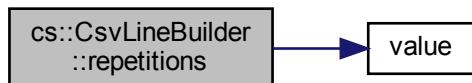
Parameters

<code>value</code>	The value to use.
--------------------	-------------------

Returns`*this`

Definition at line 106 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.9 segmentationPoints()

```
CsvLineBuilder & cs::CsvLineBuilder::segmentationPoints (
    std::uint64_t value )
```

Write accessor for the segmentation points property.

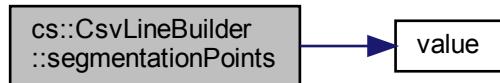
Parameters

<code>value</code>	The value to use.
--------------------	-------------------

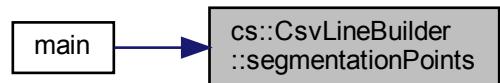
Returns`*this`

Definition at line 112 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.10 sensor()

```
CsvLineBuilder & cs::CsvLineBuilder::sensor ( std::uint64_t value )
```

Write accessor for the sensor property.

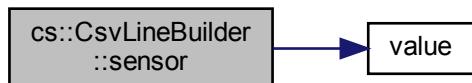
Parameters

<code>value</code>	The value to use.
--------------------	-------------------

Returns`*this`

Definition at line 100 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.11 skipWindow()

```
CsvLineBuilder & cs::CsvLineBuilder::skipWindow (
    bool value )
```

Write accessor for the skip window property.

Parameters

<code>value</code>	The value to use.
--------------------	-------------------

Returns`*this`

Definition at line 58 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.2.4.12 `windowSize()`

```
CsvLineBuilder & cs::CsvLineBuilder::windowSize( std::uint64_t value )
```

Write accessor for the window size property.

Parameters

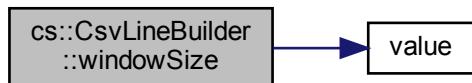
<code>value</code>	The value to use.
--------------------	-------------------

Returns

*this

Definition at line 82 of file csv_line.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



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

- compare_segmentation/include/[csv_line.hpp](#)
- compare_segmentation/src/[csv_line.cpp](#)

6.3 cl::data_set_accessor< Chan > Struct Template Reference

```
#include <channel.hpp>
```

6.3.1 Detailed Description

```
template<Channel Chan>
struct cl::data_set_accessor< Chan >
```

Definition at line 39 of file channel.hpp.

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

- [csv_lib/include/cl/channel.hpp](#)

6.4 cs::data_set_info< Tag > Struct Template Reference

Meta function for data set tags.

```
#include <data_set_info.hpp>
```

6.4.1 Detailed Description

```
template<typename Tag>
struct cs::data_set_info< Tag >
```

Meta function for data set tags.

Template Parameters

<i>Tag</i>	The data set tag to use.
------------	--------------------------

Meta function for data set tags. Contains a text for the data set tag and its repetition count.

Definition at line 21 of file data_set_info.hpp.

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

- compare_segmentation/include/[data_set_info.hpp](#)

6.5 cl::DataPoint Class Reference

```
#include <data_point.hpp>
```

Public Member Functions

- [DataPoint](#) (std::string *fileName*, long double *time*, Sensor *sensor*, Channel *channel*, long double *value*) noexcept
- const std::string & *fileName* () const noexcept
- long double *time* () const noexcept
- Sensor *sensor* () const noexcept
- Channel *channel* () const noexcept
- long double *value* () const noexcept

Friends

- std::ostream & [operator<<](#) (std::ostream &*os*, const [DataPoint](#) &*dataPoint*)

6.5.1 Detailed Description

Definition at line 10 of file data_point.hpp.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 DataPoint()

```
DataPoint::DataPoint (
    std::string fileName,
    long double time,
    Sensor sensor,
    Channel channel,
    long double value ) [noexcept]
```

Definition at line 21 of file data_point.cpp.

Here is the call graph for this function:



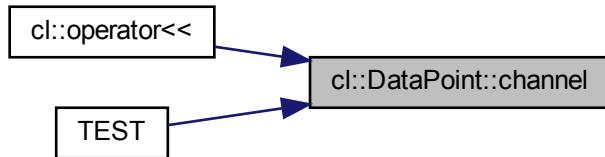
6.5.3 Member Function Documentation

6.5.3.1 channel()

```
Channel DataPoint::channel () const [noexcept]
```

Definition at line 41 of file data_point.cpp.

Here is the caller graph for this function:

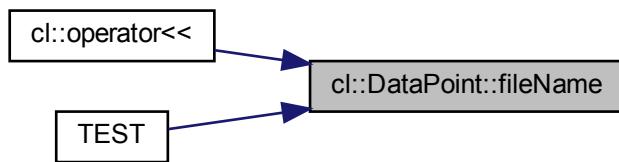


6.5.3.2 fileName()

```
const std::string & DataPoint::fileName ( ) const [noexcept]
```

Definition at line 35 of file data_point.cpp.

Here is the caller graph for this function:

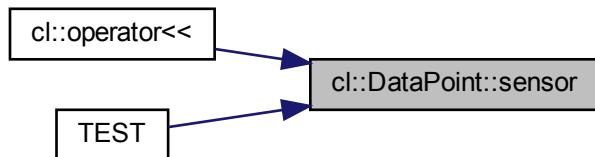


6.5.3.3 sensor()

```
Sensor DataPoint::sensor ( ) const [noexcept]
```

Definition at line 39 of file data_point.cpp.

Here is the caller graph for this function:

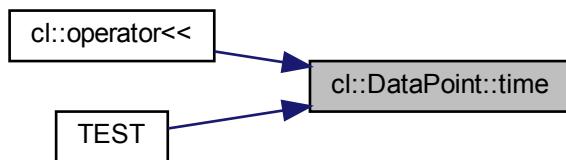


6.5.3.4 time()

```
long double DataPoint::time ( ) const [noexcept]
```

Definition at line 37 of file data_point.cpp.

Here is the caller graph for this function:

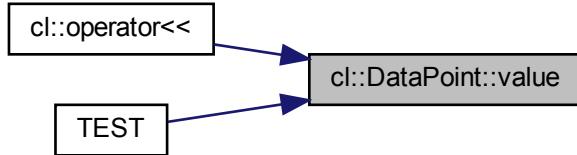


6.5.3.5 value()

```
long double DataPoint::value ( ) const [noexcept]
```

Definition at line 43 of file data_point.cpp.

Here is the caller graph for this function:



6.5.4 Friends And Related Function Documentation

6.5.4.1 operator<<

```
std::ostream& operator<< (
    std::ostream & os,
    const DataPoint & dataPoint ) [friend]
```

Definition at line 10 of file data_point.cpp.

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

- csv_lib/include/cl/data_point.hpp
- csv_lib/src/cl/data_point.cpp

6.6 cl::DataSet Class Reference

```
#include <data_set.hpp>
```

Public Types

- using `size_type` = `std::size_t`
- using `ChannelAccessor` = `long double(DataSet::*)(size_type)` const

Public Member Functions

- `size_type rowCount () const noexcept`
- `const std::string & fileName () const noexcept`
- `column_type< Column::Time > time (size_type index) const`
- `column_type< Column::HardwareTimestamp > hardwareTimestamp (size_type index) const`
- `column_type< Column::ExtractId > extractId (size_type index) const`
- `column_type< Column::Trigger > trigger (size_type index) const`
- `column_type< Column::AccelerometerX > accelerometerX (size_type index) const`
- `column_type< Column::AccelerometerY > accelerometerY (size_type index) const`
- `column_type< Column::AccelerometerZ > accelerometerZ (size_type index) const`
- `column_type< Column::GyroscopeX > gyroscopeX (size_type index) const`
- `column_type< Column::GyroscopeY > gyroscopeY (size_type index) const`
- `column_type< Column::GyroscopeZ > gyroscopeZ (size_type index) const`
- `long double accelerometerAverage (Sensor sensor) const`
- `long double gyroscopeAverage (Sensor sensor) const`
- `long double accelerometerMaximum (Sensor sensor) const`
- `long double gyroscopeMaximum (Sensor sensor) const`

Static Public Member Functions

- static `Expected< DataSet > create (std::string fileName, const std::vector< std::vector< std::string >> &matrix)`

6.6.1 Detailed Description

Definition at line 14 of file data_set.hpp.

6.6.2 Member Typedef Documentation

6.6.2.1 ChannelAccessor

```
using cl::DataSet::ChannelAccessor = long double (DataSet::*)(size_type) const
```

Definition at line 17 of file data_set.hpp.

6.6.2.2 size_type

```
using cl::DataSet::size_type = std::size_t
```

Definition at line 16 of file data_set.hpp.

6.6.3 Member Function Documentation

6.6.3.1 accelerometerAverage()

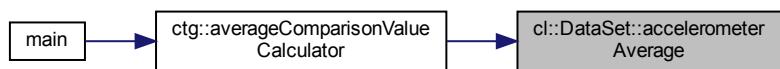
```
long double cl::DataSet::accelerometerAverage ( Sensor sensor ) const
```

Definition at line 255 of file data_set.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

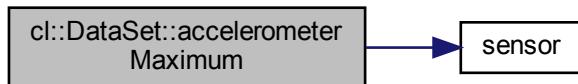


6.6.3.2 accelerometerMaximum()

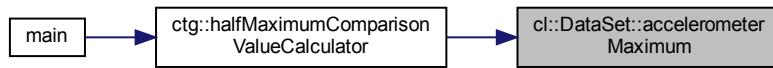
```
long double cl::DataSet::accelerometerMaximum (  
    Sensor sensor ) const
```

Definition at line 265 of file data_set.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

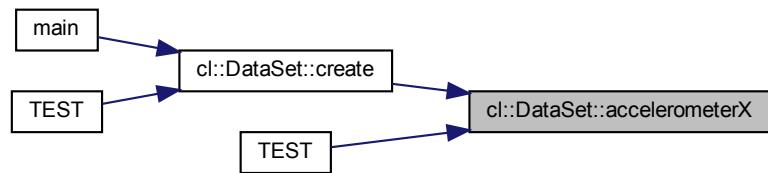


6.6.3.3 accelerometerX()

```
column_type< Column::AccelerometerX > cl::DataSet::accelerometerX (  
    size_type index ) const
```

Definition at line 200 of file data_set.cpp.

Here is the caller graph for this function:

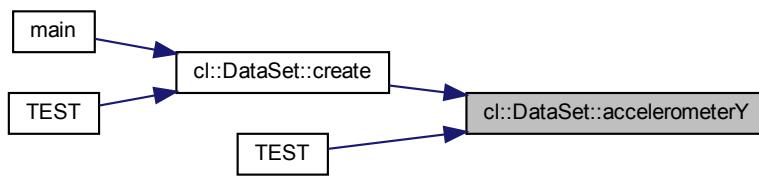


6.6.3.4 accelerometerY()

```
column_type< Column::AccelerometerY > cl::DataSet::accelerometerY ( size_type index ) const
```

Definition at line 208 of file data_set.cpp.

Here is the caller graph for this function:

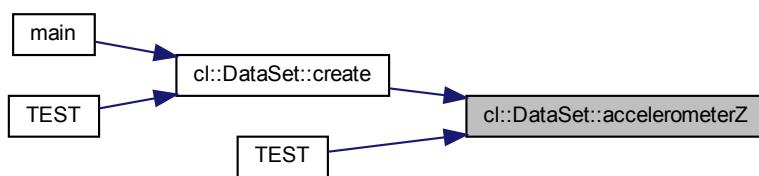


6.6.3.5 accelerometerZ()

```
column_type< Column::AccelerometerZ > cl::DataSet::accelerometerZ ( size_type index ) const
```

Definition at line 216 of file data_set.cpp.

Here is the caller graph for this function:

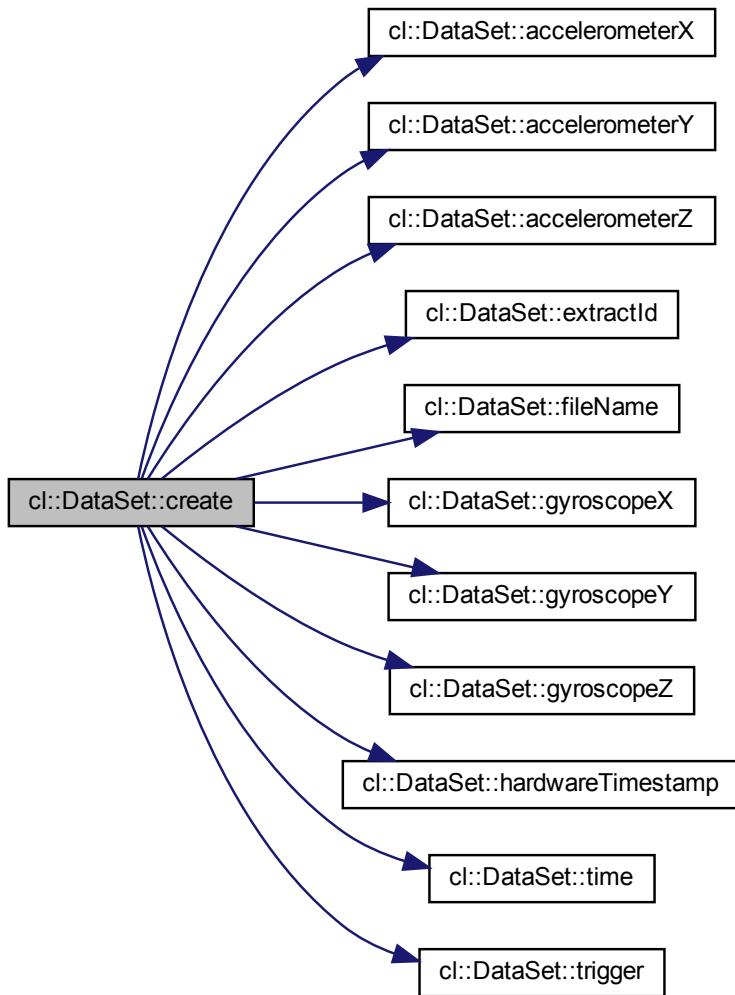


6.6.3.6 create()

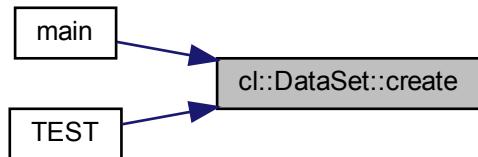
```
Expected< DataSet > cl::DataSet::create (
    std::string fileName,
    const std::vector< std::vector< std::string >> & matrix ) [static]
```

Definition at line 42 of file data_set.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

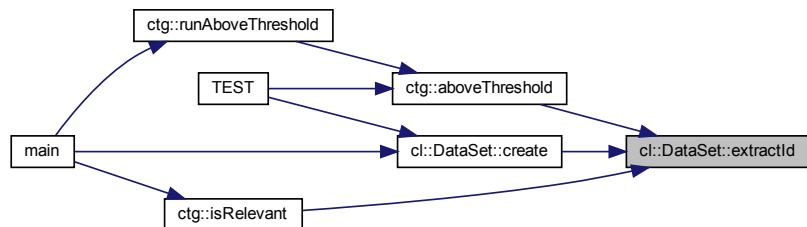


6.6.3.7 extractId()

```
column_type< Column::ExtractId > cl::DataSet::extractId (
    size_type index ) const
```

Definition at line 186 of file data_set.cpp.

Here is the caller graph for this function:

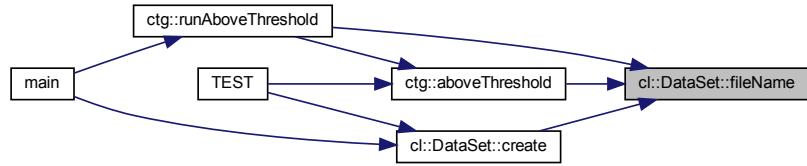


6.6.3.8 fileName()

```
const std::string & cl::DataSet::fileName ( ) const [noexcept]
```

Definition at line 169 of file data_set.cpp.

Here is the caller graph for this function:



6.6.3.9 gyroscopeAverage()

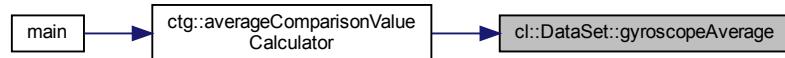
```
long double cl::DataSet::gyroscopeAverage (
    Sensor sensor ) const
```

Definition at line 260 of file data_set.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.6.3.10 gyroscopeMaximum()

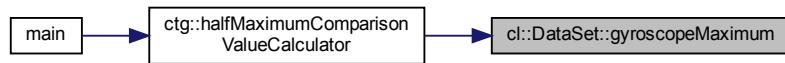
```
long double cl::DataSet::gyroscopeMaximum (
    Sensor sensor ) const
```

Definition at line 270 of file data_set.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

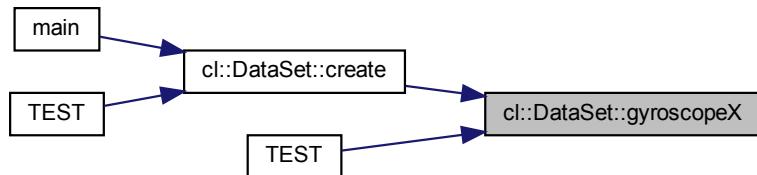


6.6.3.11 gyroscopeX()

```
column_type< Column::GyroscopeX > cl::DataSet::gyroscopeX (
    size_type index ) const
```

Definition at line 224 of file data_set.cpp.

Here is the caller graph for this function:

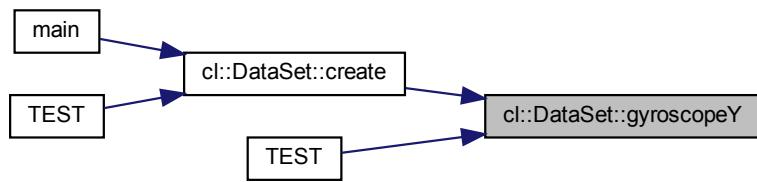


6.6.3.12 gyroscopeY()

```
column_type< Column::GyroscopeY > cl::DataSet::gyroscopeY ( size_type index ) const
```

Definition at line 231 of file data_set.cpp.

Here is the caller graph for this function:

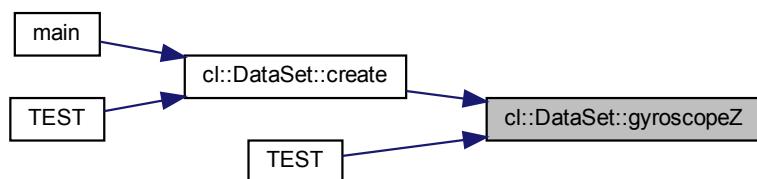


6.6.3.13 gyroscopeZ()

```
column_type< Column::GyroscopeZ > cl::DataSet::gyroscopeZ ( size_type index ) const
```

Definition at line 238 of file data_set.cpp.

Here is the caller graph for this function:

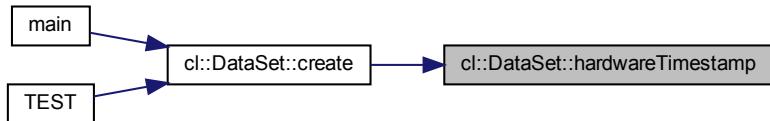


6.6.3.14 hardwareTimestamp()

```
column_type< Column::HardwareTimestamp > cl::DataSet::hardwareTimestamp (
    size_type index ) const
```

Definition at line 178 of file data_set.cpp.

Here is the caller graph for this function:

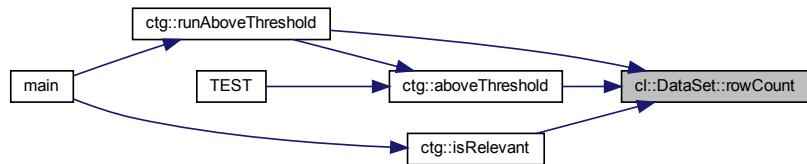


6.6.3.15 rowCount()

```
DataSet::size_type cl::DataSet::rowCount( ) const [noexcept]
```

Definition at line 152 of file data_set.cpp.

Here is the caller graph for this function:

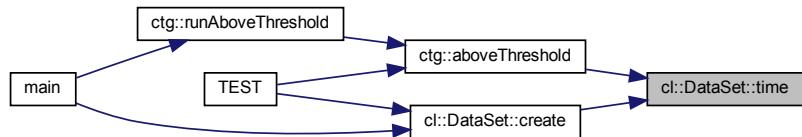


6.6.3.16 time()

```
column_type< Column::Time > cl::DataSet::time (
    size_type index ) const
```

Definition at line 171 of file data_set.cpp.

Here is the caller graph for this function:

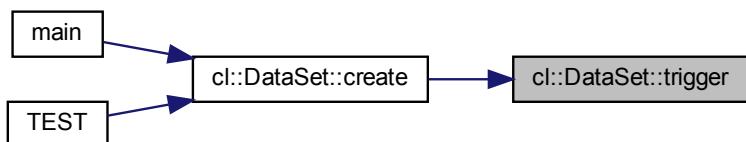


6.6.3.17 trigger()

```
column_type< Column::Trigger > cl::DataSet::trigger (
    size_type index ) const
```

Definition at line 193 of file `data_set.cpp`.

Here is the caller graph for this function:



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

- [csv_lib/include/cl/data_set.hpp](#)
- [csv_lib/src/cl/data_set.cpp](#)

6.7 cl::Error Class Reference

```
#include <error.hpp>
```

Public Types

- enum [Kind { CL_ERROR_KIND }](#)

Public Member Functions

- `Error (Kind kind, std::string file, std::string function, std::size_t line, std::string message)`
- `Kind kind () const noexcept`
- `const std::string & file () const noexcept`
- `const std::string & function () const noexcept`
- `std::size_t line () const noexcept`
- `const std::string & message () const noexcept`
- `void raise () const`
- `std::string to_string () const`

Friends

- `std::ostream & operator<< (std::ostream &os, const Error &error)`

6.7.1 Detailed Description

Definition at line 22 of file error.hpp.

6.7.2 Member Enumeration Documentation

6.7.2.1 Kind

`enum cl::Error::Kind`

Enumerator

<code>CL_ERROR_KIND</code>	<input type="button" value=""/>
----------------------------	---------------------------------

Definition at line 25 of file error.hpp.

6.7.3 Constructor & Destructor Documentation

6.7.3.1 Error()

```
cl::Error::Error (
    Kind kind,
    std::string file,
    std::string function,
    std::size_t line,
    std::string message )
```

Definition at line 41 of file error.cpp.

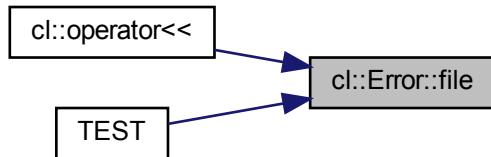
6.7.4 Member Function Documentation

6.7.4.1 file()

```
const std::string & cl::Error::file() const [noexcept]
```

Definition at line 57 of file error.cpp.

Here is the caller graph for this function:

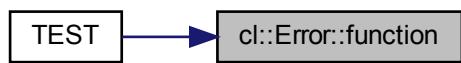


6.7.4.2 function()

```
const std::string & cl::Error::function() const [noexcept]
```

Definition at line 59 of file error.cpp.

Here is the caller graph for this function:

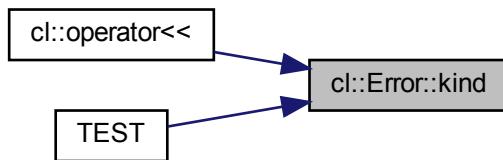


6.7.4.3 kind()

```
Error::Kind cl::Error::kind ( ) const [noexcept]
```

Definition at line 55 of file error.cpp.

Here is the caller graph for this function:



6.7.4.4 line()

```
std::size_t cl::Error::line ( ) const [noexcept]
```

Definition at line 61 of file error.cpp.

6.7.4.5 message()

```
const std::string & cl::Error::message ( ) const [noexcept]
```

Definition at line 63 of file error.cpp.

Here is the caller graph for this function:



6.7.4.6 raise()

```
void cl::Error::raise ( ) const
```

Definition at line 65 of file error.cpp.

6.7.4.7 to_string()

```
std::string cl::Error::to_string ( ) const
```

Definition at line 74 of file error.cpp.

6.7.5 Friends And Related Function Documentation

6.7.5.1 operator<<

```
std::ostream& operator<< (
    std::ostream & os,
    const Error & error ) [friend]
```

Definition at line 30 of file error.cpp.

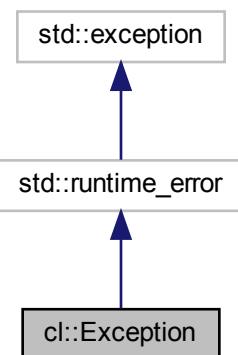
The documentation for this class was generated from the following files:

- csv_lib/include/cl/error.hpp
- csv_lib/src/cl/error.cpp

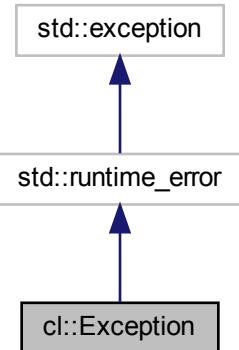
6.8 cl::Exception Class Reference

```
#include <exception.hpp>
```

Inheritance diagram for cl::Exception:



Collaboration diagram for cl::Exception:



Public Types

- using `base_type` = `std::runtime_error`

Public Member Functions

- `Exception (std::string file, std::string function, std::size_t line, const std::string &what_arg)`
- `Exception (std::string file, std::string function, std::size_t line, const char *what_arg)`
- `const std::string & file () const noexcept`
- `const std::string & function () const noexcept`
- `std::size_t line () const noexcept`

6.8.1 Detailed Description

Definition at line 14 of file exception.hpp.

6.8.2 Member Typedef Documentation

6.8.2.1 `base_type`

```
using cl::Exception::base_type = std::runtime_error
```

Definition at line 16 of file exception.hpp.

6.8.3 Constructor & Destructor Documentation

6.8.3.1 Exception() [1/2]

```
cl::Exception::Exception (
    std::string file,
    std::string function,
    std::size_t line,
    const std::string & what_arg )
```

Definition at line 6 of file exception.cpp.

6.8.3.2 Exception() [2/2]

```
cl::Exception::Exception (
    std::string file,
    std::string function,
    std::size_t line,
    const char * what_arg )
```

Definition at line 18 of file exception.cpp.

6.8.4 Member Function Documentation

6.8.4.1 file()

```
const std::string & cl::Exception::file ( ) const [noexcept]
```

Definition at line 30 of file exception.cpp.

6.8.4.2 function()

```
const std::string & cl::Exception::function ( ) const [noexcept]
```

Definition at line 32 of file exception.cpp.

6.8.4.3 line()

```
std::size_t cl::Exception::line() const [noexcept]
```

Definition at line 34 of file exception.cpp.

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

- [csv_lib/include/cl/exception.hpp](#)
- [csv_lib/src/cl/exception.cpp](#)

6.9 cl::fs::File Class Reference

Represents a file.

```
#include <file.hpp>
```

Public Member Functions

- [File \(Path path\)](#)
Creates a [File](#) from the given path.
- [bool exists \(\) const noexcept](#)
Determines if this file exists.
- [bool create \(\) const noexcept](#)
Creates this file.
- [bool copyTo \(const Path ©ToPath\) const noexcept](#)
Copies this file in the filesystem.
- [bool moveTo \(const Path &newPath\)](#)
Moves this file in the filesystem.
- [bool remove \(\) noexcept](#)
Deletes this file.
- [std::int64_t size \(\) const noexcept](#)
Determines the size of this file in bytes.
- [const Path & path \(\) const noexcept](#)
Read accessor for the path of this file.

6.9.1 Detailed Description

Represents a file.

Definition at line 11 of file file.hpp.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 File()

```
cl::fs::File::File (
    Path path ) [explicit]
```

Creates a [File](#) from the given path.

Parameters

<i>path</i>	The path to use.
-------------	------------------

Definition at line 21 of file file.cpp.

Here is the call graph for this function:



6.9.3 Member Function Documentation

6.9.3.1 copyTo()

```
bool cl::fs::File::copyTo (const Path & copyToPath) const [noexcept]
```

Copies this file in the filesystem.

Parameters

<i>copyToPath</i>	The path to copy to.
-------------------	----------------------

Returns

true if the file was successfully copied to `copyToPath`; otherwise false.

Warning

There should be no file that already exists at `copyToPath`.

Definition at line 56 of file file.cpp.

Here is the call graph for this function:



6.9.3.2 create()

```
bool cl::fs::File::create( ) const [noexcept]
```

Creates this file.

Returns

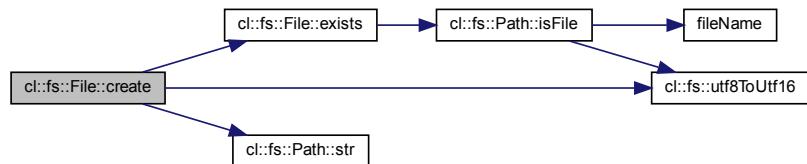
true if the file was successfully created; otherwise false.

Note

Will fail if the file already exists.

Definition at line 25 of file file.cpp.

Here is the call graph for this function:



6.9.3.3 exists()

```
bool cl::fs::File::exists () const [noexcept]
```

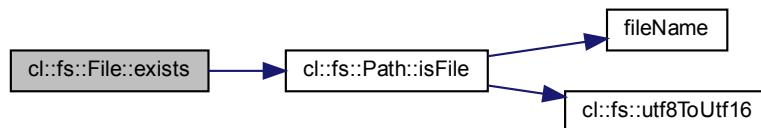
Determines if this file exists.

Returns

true if the file exists; otherwise false.

Definition at line 23 of file file.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.9.3.4 moveTo()

```
bool cl::fs::File::moveTo (
    const Path & newPath )
```

Moves this file in the filesystem.

Parameters

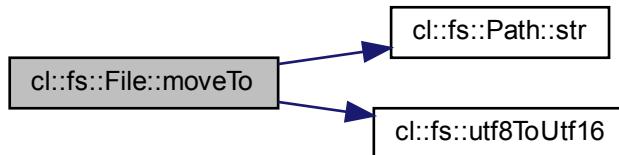
<code>newPath</code>	The path to move this file to.
----------------------	--------------------------------

Returns

true if the file was successfully moved to newPath; otherwise false.

Definition at line 100 of file file.cpp.

Here is the call graph for this function:

**6.9.3.5 path()**

```
const Path & cl::fs::File::path() const [noexcept]
```

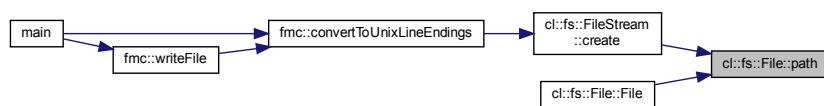
Read accessor for the path of this file.

Returns

The path of this file.

Definition at line 169 of file file.cpp.

Here is the caller graph for this function:



6.9.3.6 remove()

```
bool cl::fs::File::remove( ) [noexcept]
```

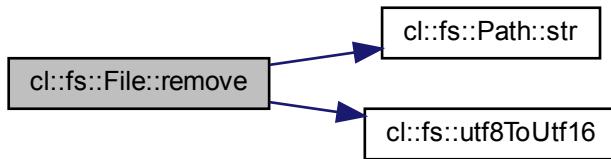
Deletes this file.

Returns

true if deleting succeeded; otherwise false.

Definition at line 117 of file file.cpp.

Here is the call graph for this function:



6.9.3.7 size()

```
std::int64_t cl::fs::File::size( ) const [noexcept]
```

Determines the size of this file in bytes.

Returns

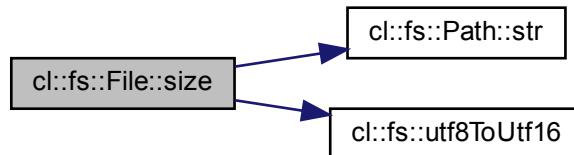
The size of this file in bytes or -1 on error.

Warning

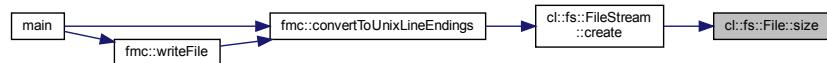
Returns -1 on error.

Definition at line 128 of file `file.cpp`.

Here is the call graph for this function:



Here is the caller graph for this function:



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

- [csv_lib/include/cl/fs/file.hpp](#)
- [csv_lib/src/cl/fs/file.cpp](#)

6.10 `cl::fs::FileStream` Class Reference

A binary file stream.

```
#include <file_stream.hpp>
```

Public Types

- enum `OpenMode` : `std::uint8_t` { `Read` = 0b0000'0001, `Write` = 0b0000'0010, `ReadWrite` = `Read` | `Write` }
The file open mode.
- using `this_type` = `FileStream`

Public Member Functions

- `PL_NONCOPYABLE (FileStream)`
- `FileStream (this_type &&other) noexcept`
Move constructs from other.
- `this_type & operator= (this_type &&other) noexcept`
Move assigns other to this file stream.
- `~FileStream ()`
Closes this file stream.
- `bool write (const void *data, std::size_t byteCount)`
Writes data to the file.
- `std::vector< pl::byte > readAll () const`
Reads the entire file into RAM.

Static Public Member Functions

- `static Expected< FileStream > create (const File &file, OpenMode openMode)`
Creates a file stream.

6.10.1 Detailed Description

A binary file stream.

Definition at line 19 of file file_stream.hpp.

6.10.2 Member Typedef Documentation

6.10.2.1 this_type

```
using cl::fs::FileStream::this_type = FileStream
```

Definition at line 30 of file file_stream.hpp.

6.10.3 Member Enumeration Documentation

6.10.3.1 OpenMode

```
enum cl::fs::FileStream::OpenMode : std::uint8_t
```

The file open mode.

Enumerator

Read	Read only access
Write	Write only access
ReadWrite	Read and write access

Definition at line 24 of file file_stream.hpp.

6.10.4 Constructor & Destructor Documentation

6.10.4.1 FileStream()

```
cl::fs::FileStream::FileStream (
    this_type && other ) [noexcept]
```

Move constructs from *other*.

Parameters

<i>other</i>	The file stream to move construct from.
--------------	---

Definition at line 70 of file file_stream.cpp.

6.10.4.2 ~FileStream()

```
cl::fs::FileStream::~FileStream ( )
```

Closes this file stream.

Definition at line 84 of file file_stream.cpp.

6.10.5 Member Function Documentation

6.10.5.1 create()

```
Expected< FileStream > cl::fs::FileStream::create (
    const File & file,
    OpenMode openMode ) [static]
```

Creates a file stream.

Parameters

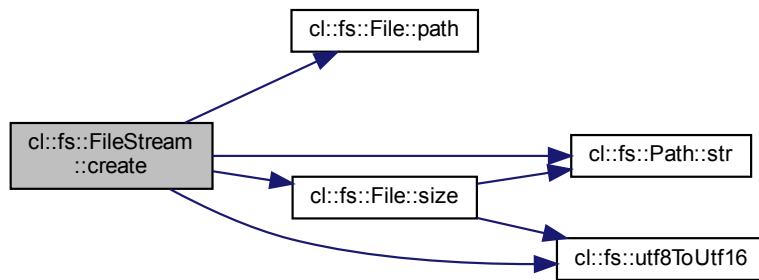
<i>file</i>	The file to open.
<i>openMode</i>	The open mode to use.

Returns

The file stream or an error.

Definition at line 36 of file file_stream.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

**6.10.5.2 operator=()**

```
FileStream & cl::fs::FileStream::operator= (
    this_type && other ) [noexcept]
```

Move assigns `other` to this file stream.

Parameters

<i>other</i>	The file stream to move assign to this file stream.
--------------	---

Returns

`*this`

Definition at line 77 of file `file_stream.cpp`.

6.10.5.3 PL_NONCOPYABLE()

```
cl::fs::FileStream::PL_NONCOPYABLE (
    FileStream )
```

6.10.5.4 readAll()

```
std::vector< pl::byte > cl::fs::FileStream::readAll ( ) const
```

Reads the entire file into RAM.

Returns

The bytes read.

Definition at line 103 of file `file_stream.cpp`.

6.10.5.5 write()

```
bool cl::fs::FileStream::write (
    const void * data,
    std::size_t byteCount )
```

Writes data to the file.

Parameters

<code>data</code>	Pointer to the beginning of the memory region to write.
<code>byteCount</code>	The amount of bytes to write, starting from <code>data</code> .

Returns

true on success; otherwise false.

Definition at line 96 of file `file_stream.cpp`.

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

- [csv_lib/include/cl/fs/file_stream.hpp](#)
- [csv_lib/src/cl/fs/file_stream.cpp](#)

6.11 cs::LogInfo Class Reference

Information about a log file.

```
#include <log_info.hpp>
```

Public Member Functions

- `LogInfo ()`
Creates an uninitialized LogInfo.
- `const cl::fs::Path & logFilePath () const noexcept`
Read accessor for the log file path.
- `bool skipWindow () const noexcept`
Read accessor for the skip window option.
- `bool deleteTooClose () const noexcept`
Read accessor for the delete too close option.
- `bool deleteLowVariance () const noexcept`
Read accessor for the delete low variance option.
- `SegmentationKind segmentationKind () const noexcept`
Read accessor for the segmentation kind.
- `std::uint64_t windowSize () const noexcept`
Read accessor for the window size.
- `FilterKind filterKind () const noexcept`
Read accessor for the filter kind.
- `std::uint64_t sensor () const noexcept`
Read accessor for the sensor.
- `bool isInitialized () const noexcept`
Checks whether this LogInfo is initialized.

Static Public Member Functions

- `static cl::Expected< LogInfo > create (cl::fs::Path logFilePath) noexcept`
Creates a LogInfo from the given log file path.

Static Public Attributes

- `static const std::uint64_t invalidSensor = UINT64_C(0xFFFFFFFFFFFFFF)`
Represents an invalid sensor.

Friends

- `bool operator== (const LogInfo &lhs, const LogInfo &rhs) noexcept`
Compares two LogInfos for equality.
- `bool operator!= (const LogInfo &lhs, const LogInfo &rhs) noexcept`
Compares two LogInfos for inequality.
- `std::ostream & operator<< (std::ostream &os, const LogInfo &logInfo)`
Prints a LogInfo to an ostream.

6.11.1 Detailed Description

Information about a log file.

Information about a log file that is extracted from the log file name.

Definition at line 20 of file log_info.hpp.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 LogInfo()

```
cs::LogInfo::LogInfo ( )
```

Creates an uninitialized [LogInfo](#).

Warning

Should only be used in order to be assigned with an initialized [LogInfo](#); otherwise use the create static member function.

Definition at line 304 of file log_info.cpp.

6.11.3 Member Function Documentation

6.11.3.1 create()

```
cl::Expected< LogInfo > cs::LogInfo::create (
    cl::fs::Path logFilePath ) [static], [noexcept]
```

Creates a [LogInfo](#) from the given log file path.

Parameters

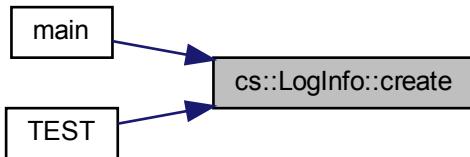
<i>logFilePath</i>	The log file path to create a LogInfo from.
--------------------	---

Returns

The [LogInfo](#) created or an error.

Definition at line 90 of file log_info.cpp.

Here is the caller graph for this function:



6.11.3.2 `deleteLowVariance()`

```
bool cs::LogInfo::deleteLowVariance () const [noexcept]
```

Read accessor for the delete low variance option.

Returns

true if delete low variance is active; false otherwise.

Definition at line 326 of file `log_info.cpp`.

6.11.3.3 `deleteTooClose()`

```
bool cs::LogInfo::deleteTooClose () const [noexcept]
```

Read accessor for the delete too close option.

Returns

true if delete too close is active; false otherwise.

Definition at line 324 of file `log_info.cpp`.

6.11.3.4 filterKind()

```
FilterKind cs::LogInfo::filterKind () const [noexcept]
```

Read accessor for the filter kind.

Returns

The filter kind.

Definition at line 335 of file log_info.cpp.

6.11.3.5 isInitialized()

```
bool cs::LogInfo::isInitialized () const [noexcept]
```

Checks whether this [LogInfo](#) is initialized.

Returns

true if this [LogInfo](#) is initialized; false otherwise.

Note

Will return true if this [LogInfo](#) was created with the create static member function.

Definition at line 339 of file log_info.cpp.

6.11.3.6 logFilePath()

```
const cl::fs::Path & cs::LogInfo::logFilePath () const [noexcept]
```

Read accessor for the log file path.

Returns

The log file path.

Definition at line 317 of file log_info.cpp.

Here is the caller graph for this function:



6.11.3.7 segmentationKind()

```
SegmentationKind cs::LogInfo::segmentationKind () const [noexcept]
```

Read accessor for the segmentation kind.

Returns

The segmentation kind.

Definition at line 328 of file log_info.cpp.

6.11.3.8 sensor()

```
std::uint64_t cs::LogInfo::sensor () const [noexcept]
```

Read accessor for the sensor.

Returns

The sensor.

Note

Will be the invalid sensor unless the log file is old.

Definition at line 337 of file log_info.cpp.

6.11.3.9 skipWindow()

```
bool cs::LogInfo::skipWindow () const [noexcept]
```

Read accessor for the skip window option.

Returns

true if skip window is active; false otherwise.

Definition at line 322 of file log_info.cpp.

6.11.3.10 `windowSize()`

```
std::uint64_t cs::LogInfo::windowSize ( ) const [noexcept]
```

Read accessor for the window size.

Returns

The window size.

Definition at line 333 of file log_info.cpp.

6.11.4 Friends And Related Function Documentation

6.11.4.1 `operator"!=`

```
bool operator!= (
    const LogInfo & lhs,
    const LogInfo & rhs ) [friend]
```

Compares two LogInfos for inequality.

Parameters

<i>lhs</i>	The left hand side operand.
<i>rhs</i>	The right hand side operand.

Returns

true if *lhs* and *rhs* are considered not equal; otherwise false.

Definition at line 287 of file log_info.cpp.

6.11.4.2 `operator<<`

```
std::ostream& operator<< (
    std::ostream & os,
    const LogInfo & logInfo ) [friend]
```

Prints a [LogInfo](#) to an ostream.

Parameters

<i>os</i>	The ostream to print to.
<i>logInfo</i>	The LogInfo to print.

Returns

`os`

Definition at line 292 of file `log_info.cpp`.

6.11.4.3 operator==

```
bool operator== (
    const LogInfo & lhs,
    const LogInfo & rhs ) [friend]
```

Compares two LogInfos for equality.

Parameters

<i>lhs</i>	The left hand side operand.
<i>rhs</i>	The right hand side operand.

Returns

true if `lhs` and `rhs` are considered equal; otherwise false.

Definition at line 264 of file `log_info.cpp`.

6.11.5 Member Data Documentation

6.11.5.1 invalidSensor

```
const std::uint64_t cs::LogInfo::invalidSensor = UINT64_C(0xFFFFFFFFFFFFFF) [static]
```

Represents an invalid sensor.

Definition at line 25 of file `log_info.hpp`.

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

- compare_segmentation/include/[log_info.hpp](#)
- compare_segmentation/src/[log_info.cpp](#)

6.12 cs::LogLine Class Reference

A line out of a log file.

```
#include <log_line.hpp>
```

Public Member Functions

- std::uint64_t [segmentationPointCount\(\)](#) const noexcept
Read accessor for the segmentation point count.
- const [cl::fs::Path & filePath\(\)](#) const noexcept
Read accessor for the file path.
- [cl::Expected< std::string > fileName\(\)](#) const
Creates the short file name for the file in the log line.
- std::uint64_t [sensor\(\)](#) const noexcept
Read accessor for the sensor.

Static Public Member Functions

- static [cl::Expected< LogLine > parse\(const std::string &line\)](#)
Parses a [LogLine](#) out of a line of text read from a log file.

Static Public Attributes

- static const std::uint64_t [invalidSensor = UINT64_C\(0xFFFFFFFFFFFFFF\)](#)
Indicates an invalid sensor.

6.12.1 Detailed Description

A line out of a log file.

Definition at line 14 of file `log_line.hpp`.

6.12.2 Member Function Documentation

6.12.2.1 `fileName()`

`cl::Expected< std::string > cs::LogLine::fileName() const`

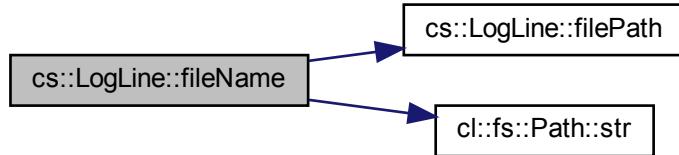
Creates the short file name for the file in the log line.

Returns

The resulting short file name or an error.

Definition at line 126 of file log_line.cpp.

Here is the call graph for this function:



6.12.2.2 filePath()

```
const cl::fs::Path & cs::LogLine::filePath ( ) const [noexcept]
```

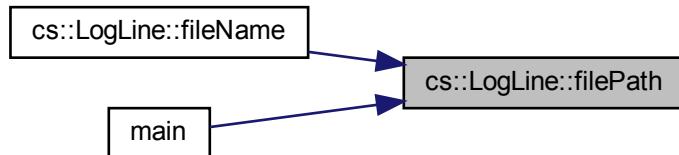
Read accessor for the file path.

Returns

The file path of the file in the log line.

Definition at line 124 of file log_line.cpp.

Here is the caller graph for this function:



6.12.2.3 parse()

```
cl::Expected< LogLine > cs::LogLine::parse (const std::string & line) [static]
```

Parses a [LogLine](#) out of a line of text read from a log file.

Parameters

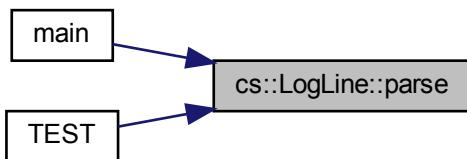
<i>line</i>	The line read.
-------------	----------------

Returns

The resulting [LogLine](#) or an error.

Definition at line 31 of file log_line.cpp.

Here is the caller graph for this function:

**6.12.2.4 segmentationPointCount()**

```
std::uint64_t cs::LogLine::segmentationPointCount ( ) const [noexcept]
```

Read accessor for the segmentation point count.

Returns

The segmentation point count.

Definition at line 119 of file log_line.cpp.

6.12.2.5 sensor()

```
std::uint64_t cs::LogLine::sensor ( ) const [noexcept]
```

Read acccessor for the sensor.

Returns

The sensor.

Note

Will only return a valid sensor if the [LogLine](#) is for a preprocessed file.

Definition at line 164 of file log_line.cpp.

6.12.3 Member Data Documentation

6.12.3.1 invalidSensor

```
const std::uint64_t cs::LogLine::invalidSensor = UINT64_C(0xFFFFFFFFFFFFFF) [static]
```

Indicates an invalid sensor.

Definition at line 19 of file log_line.hpp.

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

- compare_segmentation/include/[log_line.hpp](#)
- compare_segmentation/src/[log_line.cpp](#)

6.13 cm::ManualSegmentationPoint Class Reference

Type used to represent a manual segmentation point.

```
#include <manual_segmentation_point.hpp>
```

Public Member Functions

- [ManualSegmentationPoint](#) (std::uint32_t **hour**, std::uint32_t **minute**, std::uint32_t **second**, std::uint32_t **frame**)
Creates a ManualSegmentationPoint.
- std::uint32_t **hour** () const noexcept
Read accessor for the hour property.
- std::uint32_t **minute** () const noexcept
Read accessor for the minute property.
- std::uint32_t **second** () const noexcept
Read accessor for the second property.
- std::uint32_t **frame** () const noexcept
Read accessor for the frame property.
- std::uint64_t **asMilliseconds** () const noexcept
Converts this manual segmentation point into a millisecond representation.

Static Public Member Functions

- static std::unordered_map< [DataSetIdentifier](#), std::vector< [ManualSegmentationPoint](#) > > [readCsvFile](#) ()
Reads the CSV file of the manual segmentation points.

6.13.1 Detailed Description

Type used to represent a manual segmentation point.

Definition at line 25 of file manual_segmentation_point.hpp.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 ManualSegmentationPoint()

```
cm::ManualSegmentationPoint::ManualSegmentationPoint (
    std::uint32_t hour,
    std::uint32_t minute,
    std::uint32_t second,
    std::uint32_t frame )
```

Creates a [ManualSegmentationPoint](#).

Parameters

<i>hour</i>	The hour to use. Must be within [0,59].
<i>minute</i>	The minute to use. Must be within [0,59].
<i>second</i>	The second to use. Must be within [0,59].
<i>frame</i>	The frame to use. Must be within [0,29].

Exceptions

<i>cl::Exception</i>	if one of the arguments is out of bounds.
----------------------	---

Definition at line 270 of file manual_segmentation_point.cpp.

Here is the call graph for this function:



6.13.3 Member Function Documentation

6.13.3.1 asMilliseconds()

```
std::uint64_t cm::ManualSegmentationPoint::asMilliseconds ( ) const [noexcept]
```

Converts this manual segmentation point into a millisecond representation.

Returns

This manual segmentation point converted to milliseconds.

Definition at line 328 of file manual_segmentation_point.cpp.

6.13.3.2 frame()

```
std::uint32_t cm::ManualSegmentationPoint::frame() const [noexcept]
```

Read accessor for the frame property.

Returns

The frame within the second of this manual segmentation point.

Definition at line 323 of file manual_segmentation_point.cpp.

6.13.3.3 hour()

```
std::uint32_t cm::ManualSegmentationPoint::hour() const [noexcept]
```

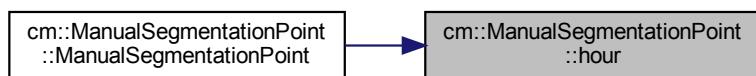
Read accessor for the hour property.

Returns

The hour.

Definition at line 311 of file manual_segmentation_point.cpp.

Here is the caller graph for this function:



6.13.3.4 minute()

```
std::uint32_t cm::ManualSegmentationPoint::minute ( ) const [noexcept]
```

Read accessor for the minute property.

Returns

The minute.

Definition at line 313 of file manual_segmentation_point.cpp.

6.13.3.5 readCsvFile()

```
std::unordered_map< DataSetIdentifier, std::vector< ManualSegmentationPoint > > cm::ManualSegmentationPoint::readCsvFile ( ) [static]
```

Reads the CSV file of the manual segmentation points.

Returns

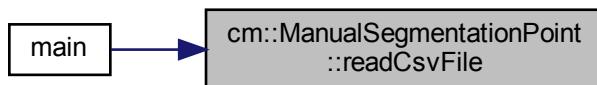
A map that maps the `DataSetIdentifier` enumerators to vectors of the corresponding manual segmentation points extracted from the CSV file.

Exceptions

<code>cl::Exception</code>	if parsing fails, CSV processing fails or the CSV file is missing.
----------------------------	--

Definition at line 128 of file manual_segmentation_point.cpp.

Here is the caller graph for this function:



6.13.3.6 second()

```
std::uint32_t cm::ManualSegmentationPoint::second ( ) const [noexcept]
```

Read accessor for the second property.

Returns

The second.

Definition at line 318 of file manual_segmentation_point.cpp.

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

- confusion_matrix/include/manual_segmentation_point.hpp
- confusion_matrix/src/manual_segmentation_point.cpp

6.14 cl::fs::Path Class Reference

A filesystem path.

```
#include <path.hpp>
```

Public Member Functions

- PL_IMPLICIT **Path** (std::string path)
Creates a path.
- bool **exists** () const noexcept
Checks if the path exists.
- bool **isFile** () const noexcept
Checks if the path is a file.
- bool **isDirectory** () const noexcept
Checks if the path is a directory.
- const std::string & **str** () const noexcept
Read accessor for the underlying string.

Friends

- std::ostream & **operator<<** (std::ostream &os, const **Path** &path)
*Prints a **Path** to an ostream.*
- bool **operator<** (const **Path** &lhs, const **Path** &rhs) noexcept
Checks if lhs is less than rhs.
- bool **operator==** (const **Path** &lhs, const **Path** &rhs) noexcept
Equality compares lhs and rhs.

6.14.1 Detailed Description

A filesystem path.

Definition at line 13 of file path.hpp.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 Path()

```
cl::fs::Path::Path (
    std::string path )
```

Creates a path.

Parameters

<i>path</i>	The string to construct from.
-------------	-------------------------------

Definition at line 37 of file path.cpp.

6.14.3 Member Function Documentation

6.14.3.1 exists()

```
bool cl::fs::Path::exists ( ) const [noexcept]
```

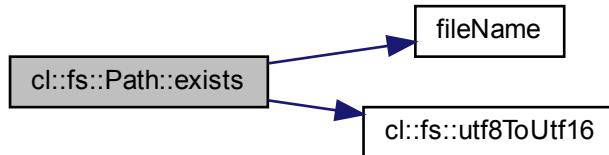
Checks if the path exists.

Returns

true if the path exists; otherwise false.

Definition at line 44 of file path.cpp.

Here is the call graph for this function:



6.14.3.2 isDirectory()

```
bool cl::fs::Path::isDirectory ( ) const [noexcept]
```

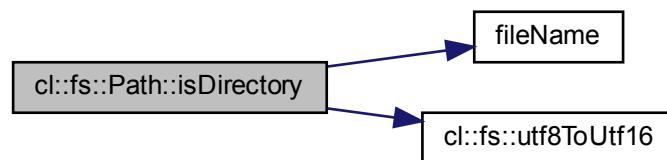
Checks if the path is a directory.

Returns

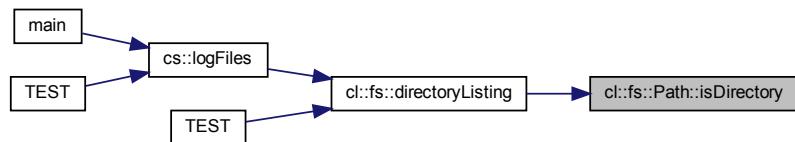
true if the path is a directory; otherwise false.

Definition at line 100 of file path.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.14.3.3 isFile()

```
bool cl::fs::Path::isFile( ) const [noexcept]
```

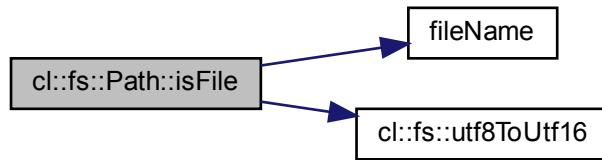
Checks if the path is a file.

Returns

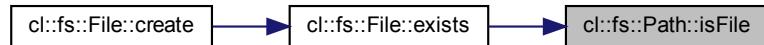
true if the path is a file; otherwise false.

Definition at line 73 of file path.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



6.14.3.4 str()

```
const std::string & cl::fs::Path::str ( ) const [noexcept]
```

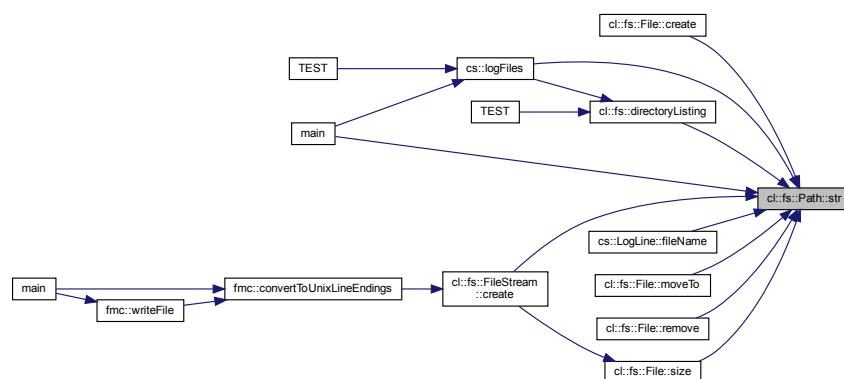
Read accessor for the underlying string.

Returns

The underlying string.

Definition at line 123 of file path.cpp.

Here is the caller graph for this function:



6.14.4 Friends And Related Function Documentation

6.14.4.1 operator<

```
bool operator< (
    const Path & lhs,
    const Path & rhs ) [friend]
```

Checks if `lhs` is less than `rhs`.

Parameters

<code>lhs</code>	The left hand side operand.
<code>rhs</code>	The right hand side operand.

Returns

true if `lhs < rhs`; otherwise false.

Definition at line 27 of file path.cpp.

6.14.4.2 operator<<

```
std::ostream& operator<< (
    std::ostream & os,
    const Path & path ) [friend]
```

Prints a [Path](#) to an ostream.

Parameters

<code>os</code>	the ostream to print to.
<code>path</code>	The path to print.

Returns

`os`

Definition at line 22 of file path.cpp.

6.14.4.3 operator==

```
bool operator== (
    const Path & lhs,
    const Path & rhs ) [friend]
```

Equality compares lhs and rhs.

Parameters

<i>lhs</i>	The left hand side operand.
<i>rhs</i>	The right hand side operand.

Returns

true if lhs and rhs are equal.

Definition at line 32 of file path.cpp.

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

- csv_lib/include/cl/fs/[path.hpp](#)
- csv_lib/src/cl/fs/[path.cpp](#)

Chapter 7

File Documentation

7.1 compare_segmentation/CMakeLists.txt File Reference

Functions

- `set (LIB_NAME compare_segmentation_lib) set(LIB_HEADERS include/csv_line.hpp include/data_set_info.hpp include/filter_kind.hpp include/log_files.hpp include/log_info.hpp include/log_line.hpp include/paths.hpp include/segmentation_kind.hpp) set(LIB_SOURCES src/csv_line.cpp src/data_set_info.cpp src/filter_kind.cpp src/log_files.cpp src/log_info.cpp src/log_line.cpp src/segmentation_kind.cpp) add_library($`

7.1.1 Function Documentation

7.1.1.1 set()

```
set (  
    LIB_NAME compare_segmentation_lib )
```

Definition at line 2 of file CMakeLists.txt.

7.2 compare_segmentation/test/CMakeLists.txt File Reference

Functions

- `include (GoogleTest) set(TEST_NAME compare_segmentation_test) set(TEST_SOURCES csv_line_test.cpp data_set_info_test.cpp log_files_test.cpp log_info_test.cpp log_line_test.cpp main.cpp) add_executable($`

7.2.1 Function Documentation

7.2.1.1 include()

```
include (
    GoogleTest    )
```

Definition at line 1 of file CMakeLists.txt.

7.3 counting/CMakeLists.txt File Reference

Functions

- `set (LIB_NAME counting_lib) set(LIB_HEADERS include/above_threshold.hpp include/average_← comparison_value_calculator.hpp include/half_maximum_comparison_value_calculator.hpp include/is_← relevant.hpp include/percentage_of.hpp include/run_above_threshold.hpp) set(LIB_SOURCES src/above← _threshold.cpp src/average_comparison_value_calculator.cpp src/half_maximum_comparison_value← calculator.cpp src/run_above_threshold.cpp) add_library($`

7.3.1 Function Documentation

7.3.1.1 set()

```
set (
    LIB_NAME counting_lib )
```

Definition at line 2 of file CMakeLists.txt.

7.4 counting/test/CMakeLists.txt File Reference

Functions

- `include (GoogleTest) set(TEST_NAME counting_test) set(TEST_SOURCES above_threshold_test.cpp main.cpp percentage_of_test.cpp) add_executable($`

7.4.1 Function Documentation

7.4.1.1 include()

```
include (
    GoogleTest    )
```

Definition at line 1 of file CMakeLists.txt.

7.5 csv_lib/CMakeLists.txt File Reference

Functions

- `set (LIB_NAME csv_lib) set(LIB_HEADERS include/cl/fs/directory_listing.hpp include/cl/fs/file.hpp include/cl/fs/file_stream.hpp include/cl/fs/path.hpp include/cl/fs/sePARATOR.hpp include/cl/fs/windows.hpp include/cl/channel.hpp include/cl/column.hpp include/cl/data_point.hpp include/cl/data_set.hpp include/cl/dos2unix.hpp include/cl/error.hpp include/cl/exception.hpp include/cl/read_csv_file.hpp include/cl/s2n.hpp include/cl/sensor.hpp include/cl/to_string.hpp include/cl/use_unbuffered_io.hpp) set(LIB_SOURCES src/cl/fs/directory_listing.cpp src/cl/fs/file.cpp src/cl/fs/file_stream.cpp src/cl/fs/path.cpp src/cl/fs/windows.hpp src/cl/channel.cpp src/cl/data_point.cpp src/cl/data_set.cpp src/cl/dos2unix.cpp src/cl/error.cpp src/cl/exception.cpp src/cl/read_csv_file.cpp src/cl/sensor.cpp src/cl/use_unbuffered_io.cpp) add_library($`

7.5.1 Function Documentation

7.5.1.1 set()

```
set (
    LIB_NAME csv_lib )
```

Definition at line 2 of file CMakeLists.txt.

7.6 csv_lib/test/CMakeLists.txt File Reference

Functions

- `include (GoogleTest) set(TEST_NAME csv_lib_test) set(TEST_SOURCES channel_test.cpp column_test.cpp data_point_test.cpp directory_listing_test.cpp error_test.cpp exception_test.cpp main.cpp sensor_test.cpp to_string_test.cpp read_csv_file_test.cpp data_set_test.cpp s2n_test.cpp) add_executable($`

7.6.1 Function Documentation

7.6.1.1 include()

```
include (
    GoogleTest )
```

Definition at line 1 of file CMakeLists.txt.

7.7 fix_csv/CMakeLists.txt File Reference

Functions

- `set (LIB_NAME fix_mogasens_csv_lib) set(LIB_HEADERS include/adjust.hardware.timestamp.hpp include/convert_to_unix_line_endings.hpp include/create_backup_file.hpp include/delete_non_bosch_sensors.hpp include/delete_out_of_bounds_values.hpp include/remove_zeros_from_field.hpp include/restore_from_backup.hpp include/write_file.hpp) set(LIB_SOURCES src/adjust.hardware.timestamp.cpp src/convert_to_unix_line_endings.cpp src/create_backup_file.cpp src/delete_non_bosch_sensors.cpp src/delete_out_of_bounds_values.cpp src/remove_zeros_from_field.cpp src/restore_from_backup.cpp src/write_file.cpp) add_library($`

7.7.1 Function Documentation

7.7.1.1 set()

```
set (
    LIB_NAME fix_mogasens_csv_lib )
```

Definition at line 2 of file CMakeLists.txt.

7.8 fix_csv/test/CMakeLists.txt File Reference

Functions

- `include (GoogleTest) set(TEST_NAME fmc_test) set(TEST_SOURCES main.cpp remove_zeros_from_field_test.cpp adjust.hardware.timestamp_test.cpp) add_executable($`

7.8.1 Function Documentation

7.8.1.1 include()

```
include (
    GoogleTest )
```

Definition at line 1 of file CMakeLists.txt.

7.9 confusion_matrix/CMakeLists.txt File Reference

Functions

- `set (LIB_NAME confusion_matrix_lib) set(LIB_HEADERS include/data_set_identifier.hpp include/manual_segmentation_point.hpp) set(LIB_SOURCES src/manual_segmentation_point.cpp) add_library($`

7.9.1 Function Documentation

7.9.1.1 set()

```
set (
    LIB_NAME confusion_matrix_lib )
```

Definition at line 2 of file CMakeLists.txt.

7.10 confusion_matrix/test/CMakeLists.txt File Reference

Functions

- `include (GoogleTest) set(TEST_NAME confusion_matrix_test) set(TEST_SOURCES main.cpp manual_segmentation_point_test.cpp) add_executable($`

7.10.1 Function Documentation

7.10.1.1 include()

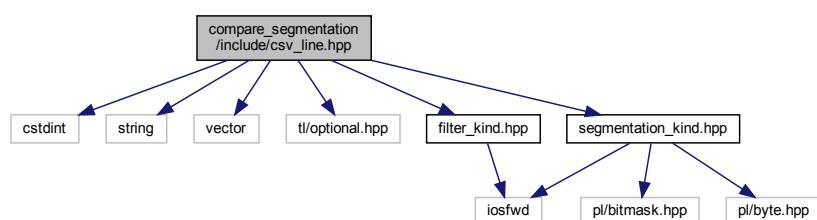
```
include (
    GoogleTest )
```

Definition at line 1 of file CMakeLists.txt.

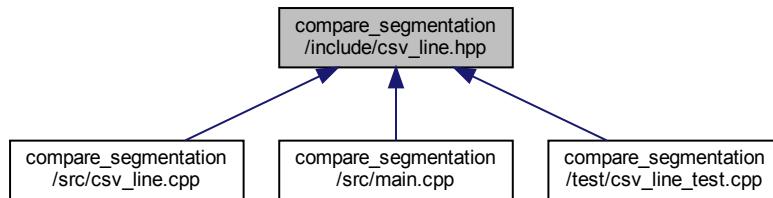
7.11 compare_segmentation/include/csv_line.hpp File Reference

```
#include <cstdint>
#include <string>
#include <vector>
#include <tl/optional.hpp>
#include "filter_kind.hpp"
#include "segmentation_kind.hpp"
```

Include dependency graph for csv_line.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [cs::CsvLineBuilder](#)

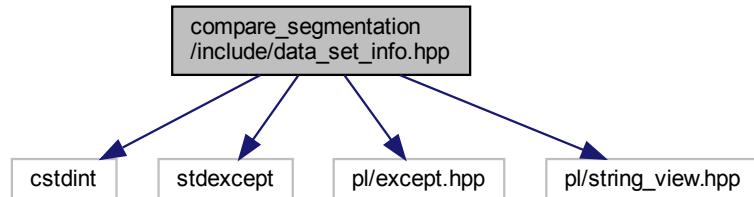
Builder for a CSV line.

Namespaces

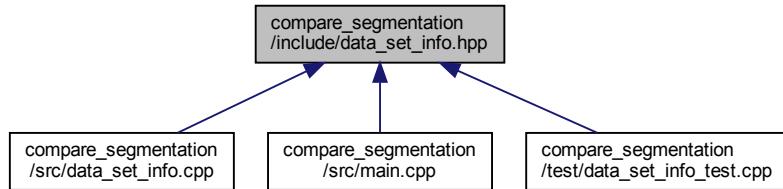
- [cs](#)

7.12 compare_segmentation/include/data_set_info.hpp File Reference

```
#include <cstdint>
#include <stdexcept>
#include <pl/except.hpp>
#include <pl/string_view.hpp>
Include dependency graph for data_set_info.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct `cs::data_set_info< Tag >`
Meta function for data set tags.

Namespaces

- `cs`

Macros

- `#define CS_SPECIALIZE_DATA_SET_INFO(tag, string, repetitionCount)`

Functions

- `cs::PL_DEFINE_EXCEPTION_TYPE` (NoSuchDataSetException, std::logic_error)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Felix1, "11.17.39", 24)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Felix2, "12.50.00", 20)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Felix3, "13.00.09", 15)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Marcelle1, "14.59.59", 10)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Marcelle2, "15.13.22", 16)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Marcelle3, "15.31.36", 18)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Mike1, "14.07.33", 26)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Mike2, "14.14.32", 22)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Mike3, "14.20.28", 18)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Andre1, "Andre_liegestuetzen1", 27)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Andre2, "Andre_liegestuetzen2", 20)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Andre3, "Andre_liegestuetzen3", 17)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (AndreSquats1, "Andre_Squats", 30)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (AndreSquats2, "Andre_Squats2", 49)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Jan1, "Jan_liegestuetzen1", 25)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Jan2, "Jan_liegestuetzen2", 19)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Jan3, "Jan_liegestuetzen3", 13)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Lucas1, "Lucas_liegestuetzen1", 24)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Lucas2, "Lucas_liegestuetzen2", 19)
- `cs::CS_SPECIALIZE_DATA_SET_INFO` (Lucas3, "Lucas_liegestuetzen3", 11)
- `std::uint64_t cs::repetitionCount (pl::string_view dataSet)`
Fetches the repetition count for a given data set identified by its string.

7.12.1 Macro Definition Documentation

7.12.1.1 CS_SPECIALIZE_DATA_SET_INFO

```
#define CS_SPECIALIZE_DATA_SET_INFO(
    tag,
    string,
    repetitionCount )
```

Value:

```
struct tag {
};

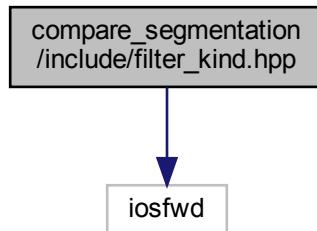
constexpr bool contains##tag(pl::string_view other)
{
    return other.contains(string);
}

template<tag>
struct data_set_info<tag> {
    static constexpr pl::string_view text      = string;
    static constexpr std::uint64_t repetitions = UINT64_C(repetitionCount); \
}
```

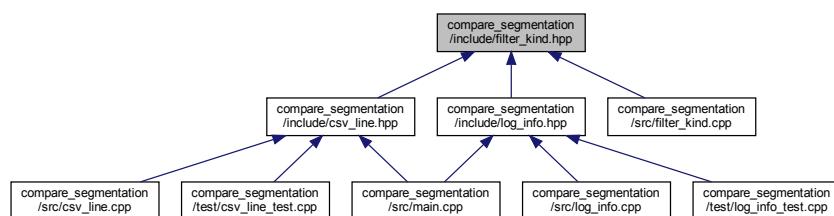
Definition at line 23 of file data_set_info.hpp.

7.13 compare_segmentation/include/filter_kind.hpp File Reference

```
#include <iostream>
Include dependency graph for filter_kind.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

- CS

Enumerations

- enum cs::FilterKind { cs::FilterKind::Butterworth, cs::FilterKind::MovingAverage }

Type for the different kinds of filters.

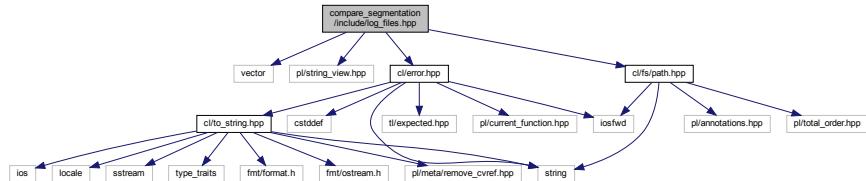
Functions

- std::ostream & `cs::operator<<` (std::ostream &os, FilterKind filterKind)

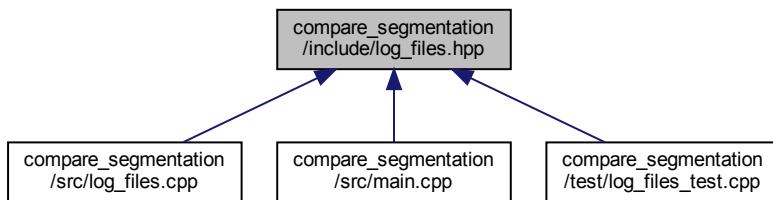
Prints a FilterKind to an ostream.

7.14 compare_segmentation/include/log_files.hpp File Reference

```
#include <vector>
#include <pl/string_view.hpp>
#include <c1/error.hpp>
#include <c1/fs/path.hpp>
Include dependency graph for log_files.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

- CS

Functions

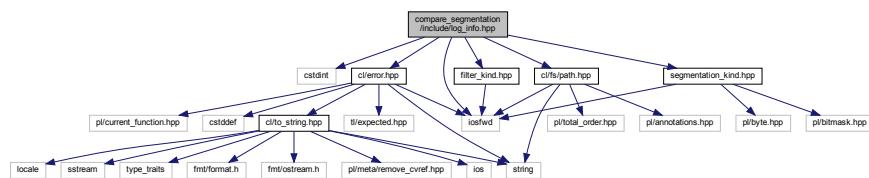
- `cl::Expected< std::vector< cl::fs::Path > > cs::logFiles (pl::string_view directoryPath)`

Fetches the paths to the log files in the given directory.

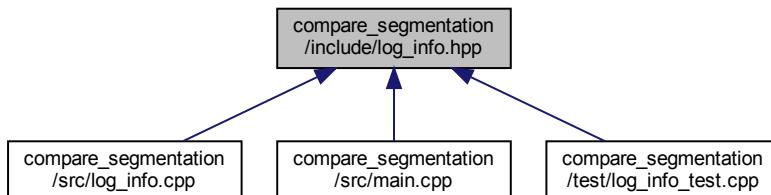
7.15 compare_segmentation/include/log_info.hpp File Reference

```
#include <cstdint>
#include <iostream>
#include <cl/error.hpp>
#include <cl/fs/path.hpp>
#include "filter_kind.hpp"
#include "segmentation_kind.hpp"
```

Include dependency graph for log_info.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class `cs::LogInfo`

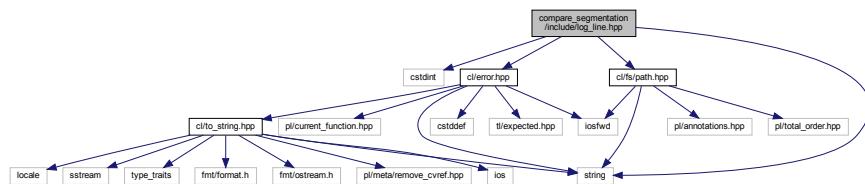
Information about a log file.

Namespaces

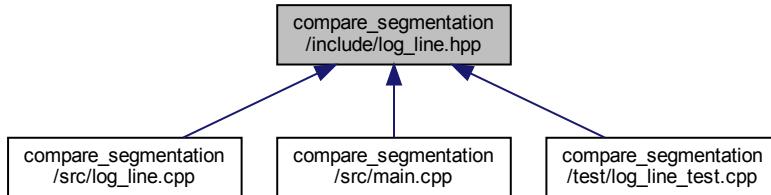
- `CS`

7.16 compare_segmentation/include/log_line.hpp File Reference

```
#include <cstdint>
#include <string>
#include "cl/error.hpp"
#include "cl/fs/path.hpp"
Include dependency graph for log_line.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [cs::LogLine](#)

A line out of a log file.

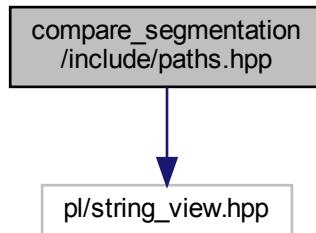
Namespaces

- [cs](#)

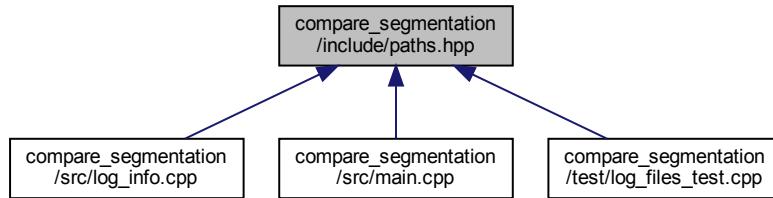
7.17 compare_segmentation/include/paths.hpp File Reference

```
#include <pl/string_view.hpp>
```

Include dependency graph for paths.hpp:



This graph shows which files directly or indirectly include this file:



Namespaces

- [cs](#)

Variables

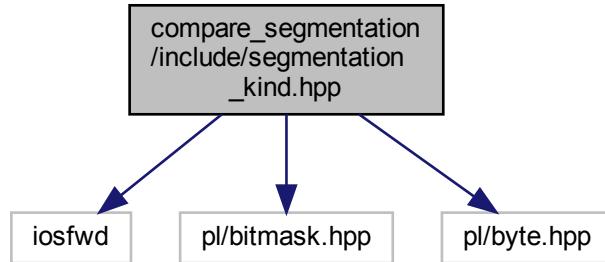
- `constexpr pl::string_view cs::logPath {"segmentation_comparison/logs"}`
Relative path to the directory containing the preprocessed log files.
- `constexpr pl::string_view cs::oldLogPath {"segmentation_comparison/logs/old"}`
Relative path to the directory containing the old log files.

7.18 compare_segmentation/include/segmentation_kind.hpp File Reference

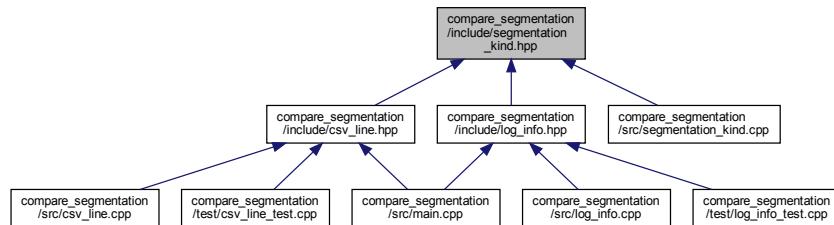
```
#include <iostream>
#include <pl/bitmask.hpp>
```

```
#include <pl/byte.hpp>
```

Include dependency graph for segmentation_kind.hpp:



This graph shows which files directly or indirectly include this file:



Namespaces

- `cs`

Enumerations

- enum `cs::SegmentationKind` : `pl::byte` { `cs::SegmentationKind::Minima` = `0b0000'0001`, `cs::SegmentationKind::Maxima` = `0b0000'0010`, `cs::SegmentationKind::Both` = `Minima | Maxima` }

The segmentation kind.

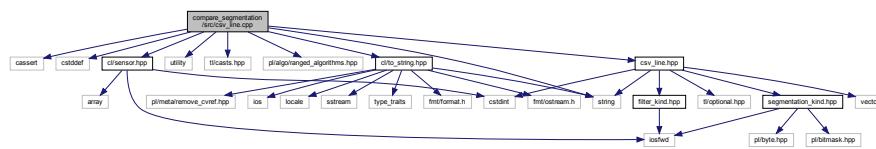
Functions

- `std::ostream & cs::operator<< (std::ostream &os, SegmentationKind segmentationKind)`
Prints a SegmentationKind to an ostream.

7.19 compare_segmentation/src/csv_line.cpp File Reference

```
#include <cassert>
#include <cstddef>
#include <string>
#include <utility>
#include <tl/casts.hpp>
#include <pl/algo/ranged_algorithms.hpp>
#include "cl/sensor.hpp"
#include "cl/to_string.hpp"
#include "csv_line.hpp"

Include dependency graph for csv_line.cpp:
```



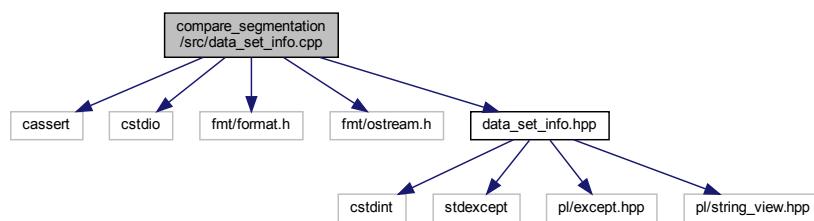
Namespaces

- [CS](#)

7.20 compare_segmentation/src/data_set_info.cpp File Reference

```
#include <cassert>
#include <cstdio>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include "data_set_info.hpp"

Include dependency graph for data_set_info.cpp:
```



Namespaces

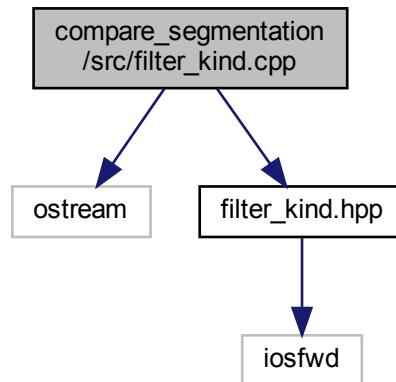
- [CS](#)

Functions

- std::uint64_t [cs::repetitionCount](#) (pl::string_view dataSet)
Fetches the repetition count for a given data set identified by its string.

7.21 compare_segmentation/src/filter_kind.cpp File Reference

```
#include <iostream>
#include "filter_kind.hpp"
Include dependency graph for filter_kind.cpp:
```



Namespaces

- [cs](#)

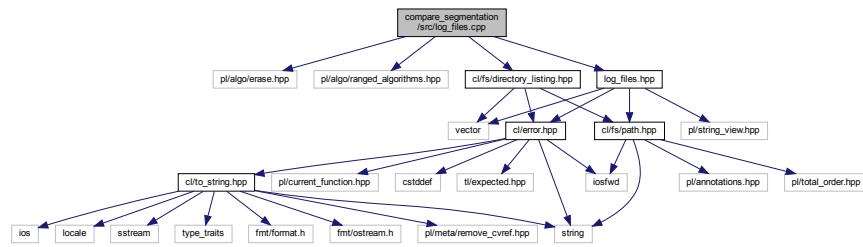
Functions

- std::ostream & [cs::operator<<](#) (std::ostream &os, FilterKind filterKind)
Prints a FilterKind to an ostream.

7.22 compare_segmentation/src/log_files.cpp File Reference

```
#include <pl/algo/erase.hpp>
#include <pl/algo/ranged_algorithms.hpp>
#include <cl/fs/directory_listing.hpp>
```

```
#include "log_files.hpp"
Include dependency graph for log_files.cpp
```



Namespaces

- CS

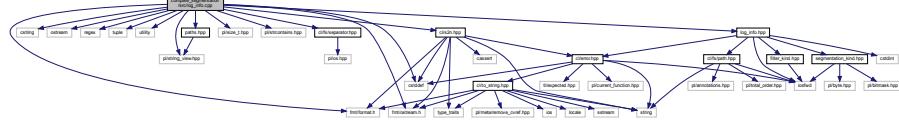
Functions

- `cl::Expected< std::vector< cl::fs::Path > > cs::logFiles (pl::string_view directoryPath)`
Fetches the paths to the log files in the given directory.

7.23 compare_segmentation/src/log_info.cpp File Reference

```
#include <cstddef>
#include <cstring>
#include <iostream>
#include <regex>
#include <tuple>
#include <utility>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/size_t.hpp>
#include <pl/strcontains.hpp>
#include <pl/string_view.hpp>
#include "cl/fs/sePARATOR.hpp"
#include "cl/s2n.hpp"
#include "log_info.hpp"
#include "paths.hpp"
```

Include dependency graph for log_info.cpp.



Namespaces

- cs

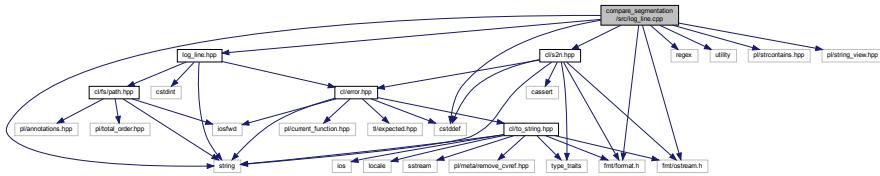
Functions

- bool `cs::operator==` (const LogInfo &lhs, const LogInfo &rhs) noexcept
 - bool `cs::operator!=` (const LogInfo &lhs, const LogInfo &rhs) noexcept
 - std::ostream & `cs::operator<<` (std::ostream &os, const LogInfo &logInfo)

7.24 compare_segmentation/src/log_line.cpp File Reference

```
#include <cstddef>
#include <regex>
#include <string>
#include <utility>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/strcontains.hpp>
#include <pl/string_view.hpp>
#include "cl/s2n.hpp"
#include "log_line.hpp"
Include dependency graph for log_line.cpp:
```

Include dependency graph for log_line.cpp:



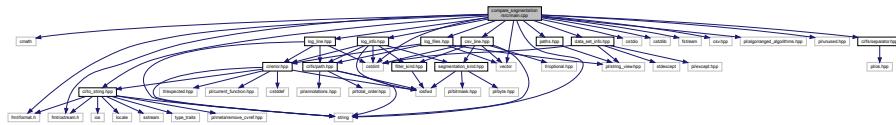
Namespaces

- CS

7.25 compare_segmentation/src/main.cpp File Reference

```
#include <cmath>
#include <cstdint>
#include <cstdio>
#include <cstdlib>
#include <fstream>
#include <string>
#include <vector>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <csv.hpp>
#include <pl/algo/ranged_algorithms.hpp>
#include <pl/unused.hpp>
#include "cl/fs/sePARATOR.hpp"
#include "cl/to_string.hpp"
#include "CSV_line.hpp"
#include "data_set_info.hpp"
#include "log_files.hpp"
```

```
#include "log_info.hpp"  
#include "log_line.hpp"  
#include "paths.hpp"  
Include dependency graph for main.cpp:
```



Functions

- int main (int argc, char *argv[])

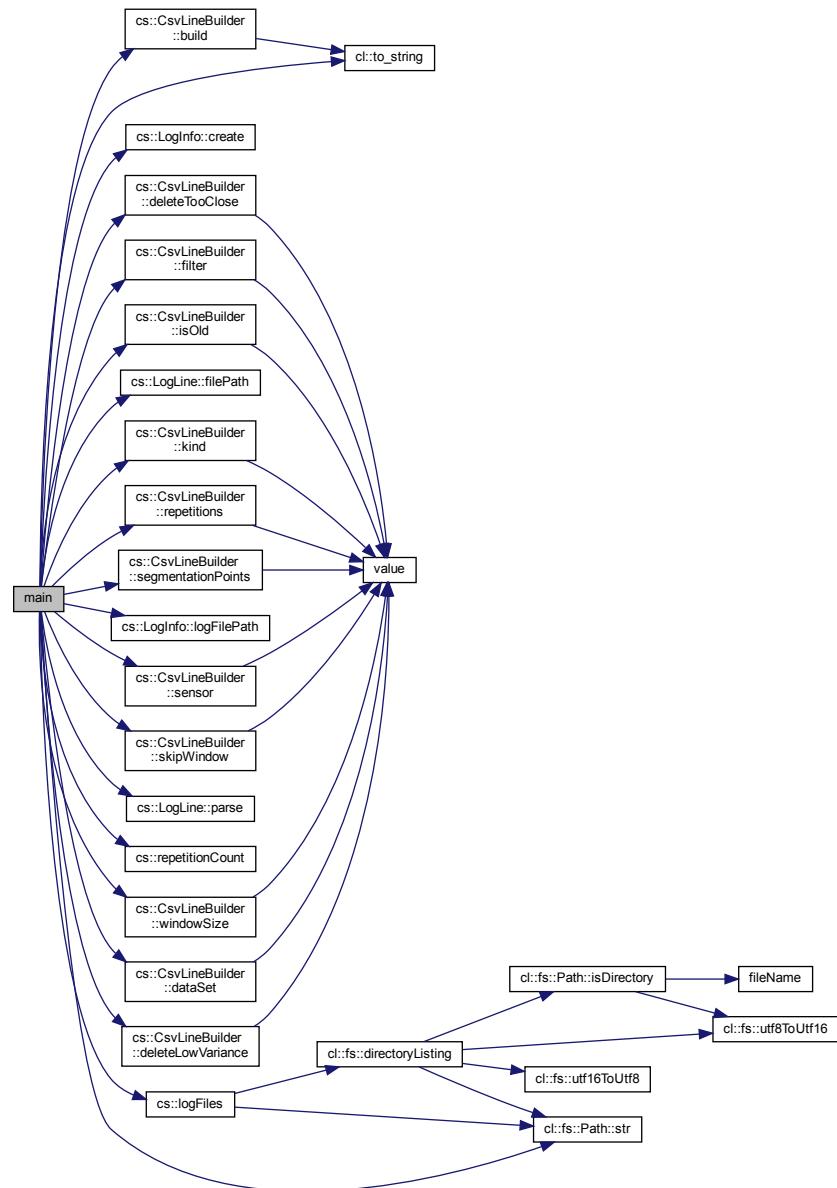
7.25.1 Function Documentation

7.25.1.1 main()

```
int main ( int argc,  
           char * argv[ ] )
```

Definition at line 28 of file main.cpp.

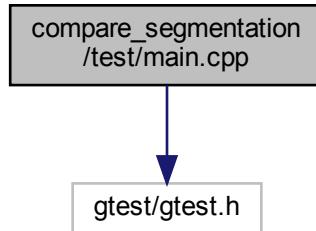
Here is the call graph for this function:



7.26 compare_segmentation/test/main.cpp File Reference

```
#include "gtest/gtest.h"
```

Include dependency graph for main.cpp:



Functions

- int [main](#) (int argc, char *argv[])

7.26.1 Function Documentation

7.26.1.1 [main\(\)](#)

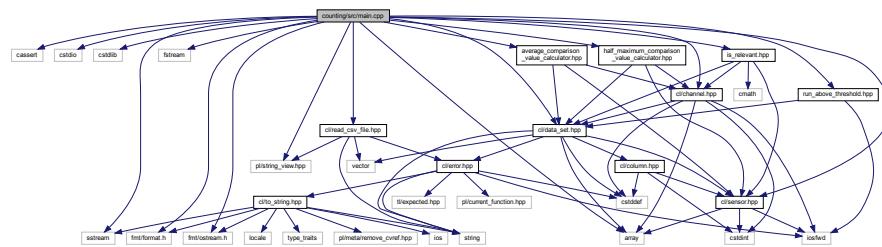
```
int main (
    int argc,
    char * argv[] )
```

Definition at line 3 of file main.cpp.

7.27 counting/src/main.cpp File Reference

```
#include <cassert>
#include <cstdio>
#include <cstdlib>
#include <array>
#include <fstream>
#include <sstream>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/string_view.hpp>
#include "cl/channel.hpp"
#include "cl/data_set.hpp"
#include "cl/read_csv_file.hpp"
#include "cl/sensor.hpp"
#include "average_comparison_value_calculator.hpp"
#include "half_maximum_comparison_value_calculator.hpp"
```

```
#include "is_relevant.hpp"
#include "run_above_threshold.hpp"
Include dependency graph for main.cpp:
```



Functions

- int main (int argc, char *argv[])

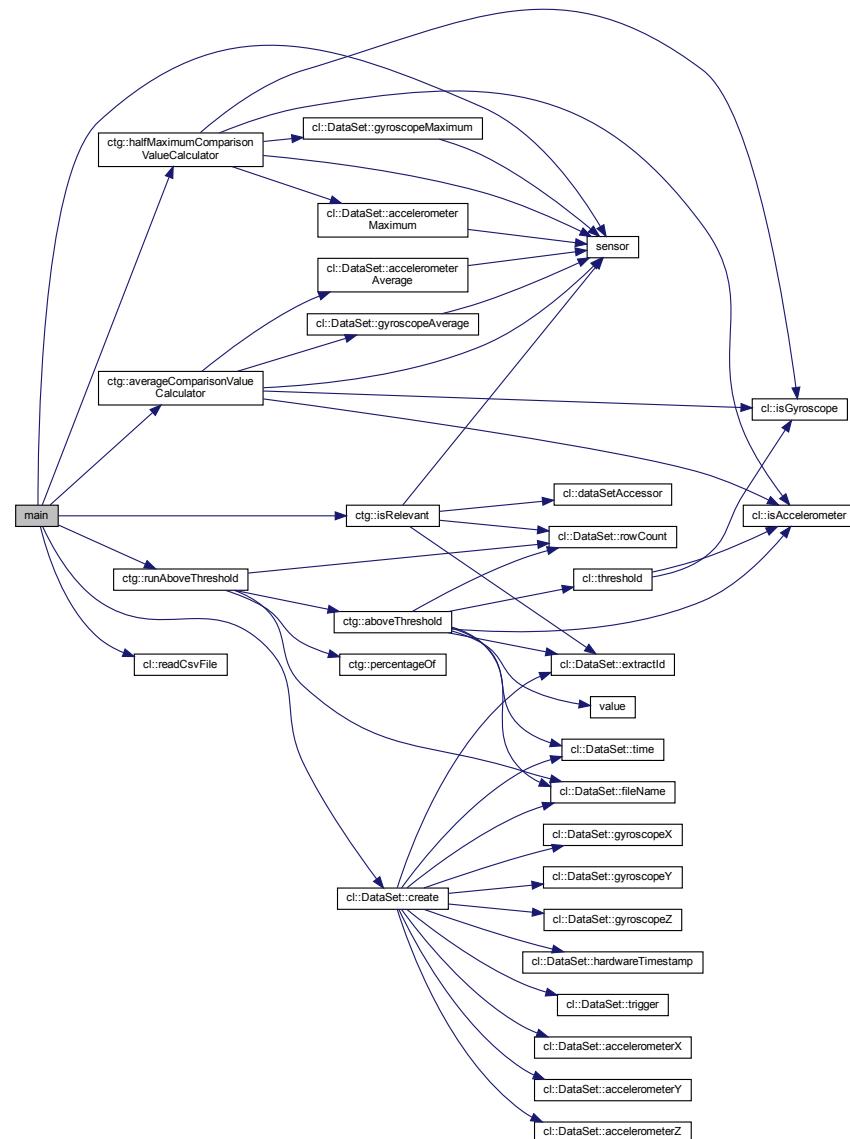
7.27.1 Function Documentation

7.27.1.1 main()

```
int main ( int argc,  
           char * argv[ ] )
```

Definition at line 24 of file main.cpp.

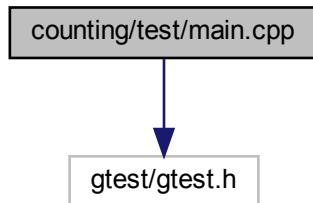
Here is the call graph for this function:



7.28 counting/test/main.cpp File Reference

```
#include "gtest/gtest.h"
```

Include dependency graph for main.cpp:



Functions

- int `main` (int argc, char *argv[])

7.28.1 Function Documentation

7.28.1.1 main()

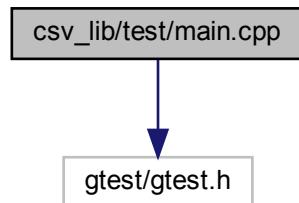
```
int main (
    int argc,
    char * argv[] )
```

Definition at line 3 of file main.cpp.

7.29 csv_lib/test/main.cpp File Reference

```
#include "gtest/gtest.h"
```

Include dependency graph for main.cpp:



Functions

- int main (int argc, char *argv[])

7.29.1 Function Documentation

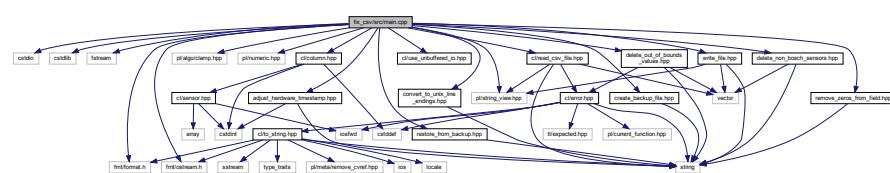
7.29.1.1 main()

```
int main ( int argc,  
           char * argv[ ] )
```

Definition at line 3 of file main.cpp.

7.30 fix_csv/src/main.cpp File Reference

```
#include <cstdio>
#include <cstdlib>
#include <fstream>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/algo/clamp.hpp>
#include <pl/numeric.hpp>
#include <pl/string_view.hpp>
#include "cl/column.hpp"
#include "cl/read_csv_file.hpp"
#include "cl/use_unbuffered_io.hpp"
#include "adjust_hardware_timestamp.hpp"
#include "convert_to_unix_line_endings.hpp"
#include "create_backup_file.hpp"
#include "delete_non_bosch_sensors.hpp"
#include "delete_out_of_bounds_values.hpp"
#include "remove_zeros_from_field.hpp"
#include "restore_from_backup.hpp"
#include "write_file.hpp"
Include dependency graph for main.cpp:
```



Functions

- int main (int argc, char *argv[])

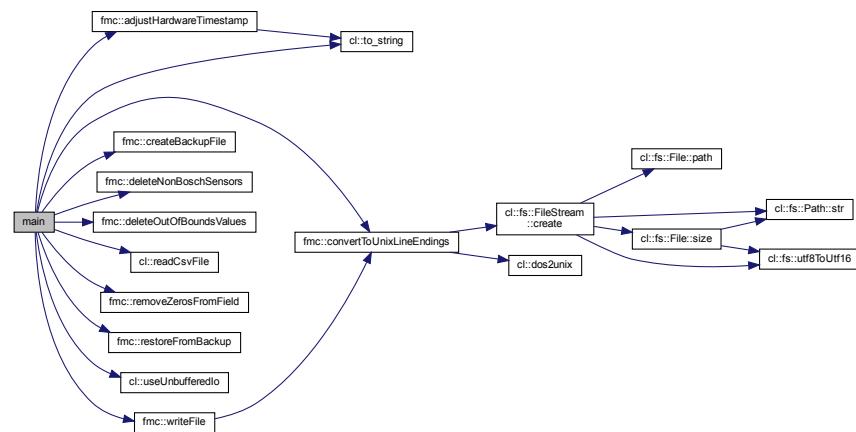
7.30.1 Function Documentation

7.30.1.1 main()

```
int main (
    int argc,
    char * argv[] )
```

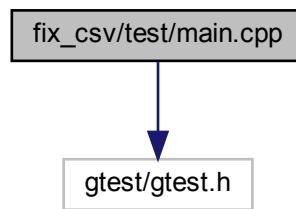
Definition at line 26 of file main.cpp.

Here is the call graph for this function:



7.31 fix_csv/test/main.cpp File Reference

```
#include "gtest/gtest.h"
Include dependency graph for main.cpp:
```



Functions

- int [main](#) (int argc, char *argv[])

7.31.1 Function Documentation

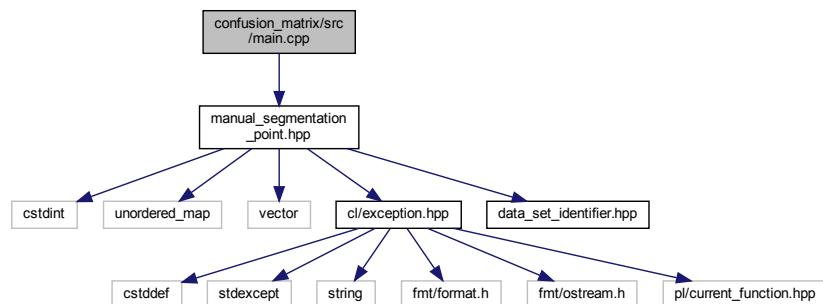
7.31.1.1 main()

```
int main (
    int argc,
    char * argv[ ] )
```

Definition at line 3 of file main.cpp.

7.32 confusion_matrix/src/main.cpp File Reference

```
#include "manual_segmentation_point.hpp"
Include dependency graph for main.cpp:
```



Functions

- int [main](#) ()

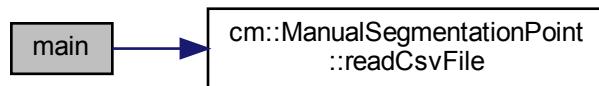
7.32.1 Function Documentation

7.32.1.1 main()

```
int main ( )
```

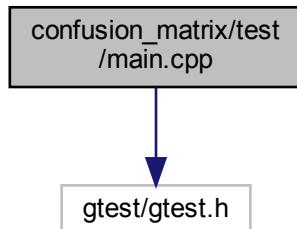
Definition at line 3 of file main.cpp.

Here is the call graph for this function:



7.33 confusion_matrix/test/main.cpp File Reference

```
#include "gtest/gtest.h"  
Include dependency graph for main.cpp:
```



Functions

- int `main` (int argc, char *argv[])

7.33.1 Function Documentation

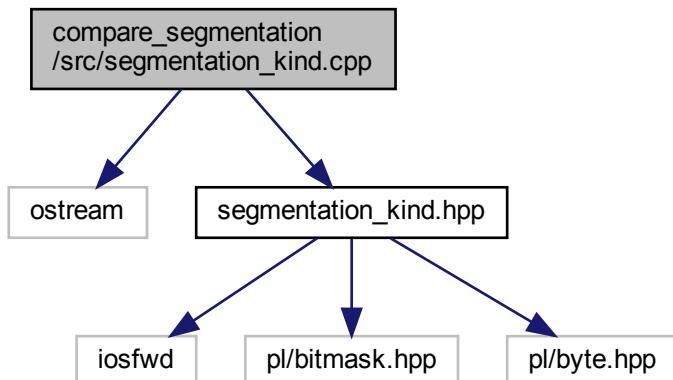
7.33.1.1 main()

```
int main (
    int argc,
    char * argv[] )
```

Definition at line 3 of file main.cpp.

7.34 compare_segmentation/src/segmentation_kind.cpp File Reference

```
#include <iostream>
#include "segmentation_kind.hpp"
Include dependency graph for segmentation_kind.cpp:
```



Namespaces

- `cs`

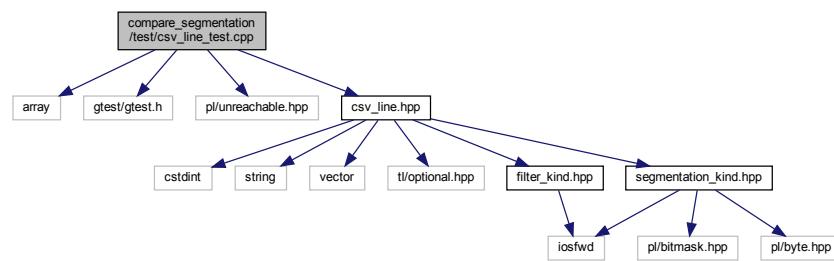
Functions

- `std::ostream & cs::operator<< (std::ostream &os, SegmentationKind segmentationKind)`

Prints a SegmentationKind to an ostream.

7.35 compare_segmentation/test/csv_line_test.cpp File Reference

```
#include <array>
#include "gtest/gtest.h"
#include <pl/unreachable.hpp>
#include "csv_line.hpp"
Include dependency graph for csv_line_test.cpp:
```



Functions

- [TEST](#) (CsvLine, shouldWork)

7.35.1 Function Documentation

7.35.1.1 TEST()

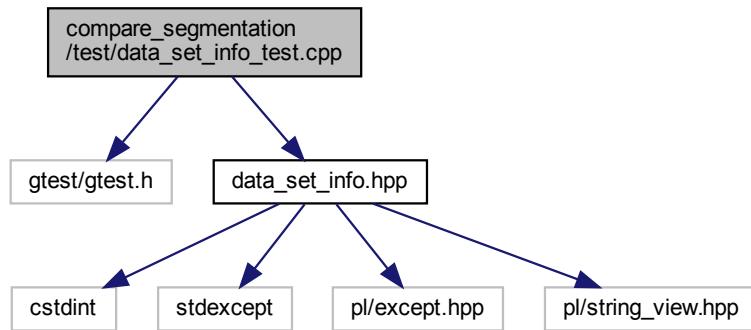
```
TEST (
    CsvLine ,
    shouldWork )
```

Definition at line 30 of file csv_line_test.cpp.

7.36 compare_segmentation/test/data_set_info_test.cpp File Reference

```
#include "gtest/gtest.h"
#include "data_set_info.hpp"
```

Include dependency graph for data_set_info_test.cpp:



Functions

- [TEST](#) (dataSetInfo, repetitionCount)

7.36.1 Function Documentation

7.36.1.1 TEST()

```
TEST (
    dataSetInfo ,
    repetitionCount )
```

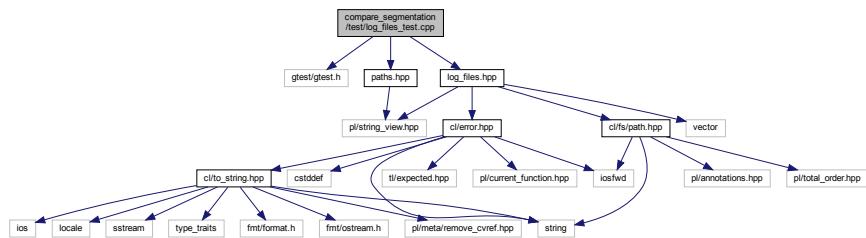
Definition at line 5 of file `data_set_info_test.cpp`.

Here is the call graph for this function:



7.37 compare_segmentation/test/log_files_test.cpp File Reference

```
#include "gtest/gtest.h"
#include <log_files.hpp>
#include <paths.hpp>
Include dependency graph for log_files_test.cpp:
```



Functions

- [TEST](#) (logFiles, shouldFindLogFiles)
- [TEST](#) (logFiles, shouldFindOldLogFiles)
- [TEST](#) (logFiles, shouldNotFindGarbage)

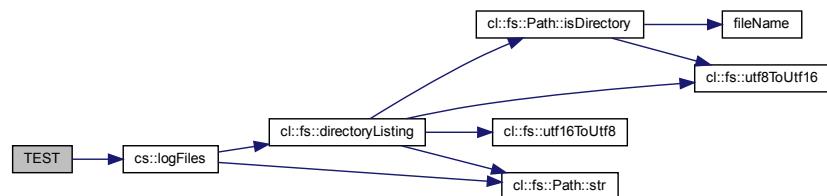
7.37.1 Function Documentation

7.37.1.1 TEST() [1/3]

```
TEST (
    logFiles ,
    shouldFindLogFiles )
```

Definition at line 6 of file log_files_test.cpp.

Here is the call graph for this function:

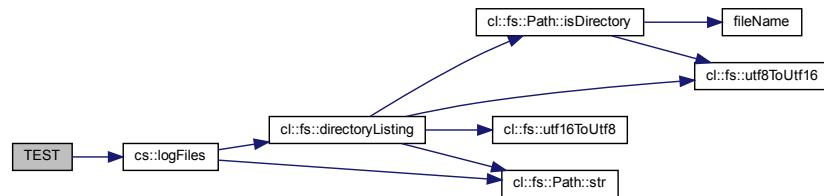


7.37.1.2 TEST() [2/3]

```
TEST (
    logFiles ,
    shouldFindOldLogFiles )
```

Definition at line 23 of file log_files_test.cpp.

Here is the call graph for this function:

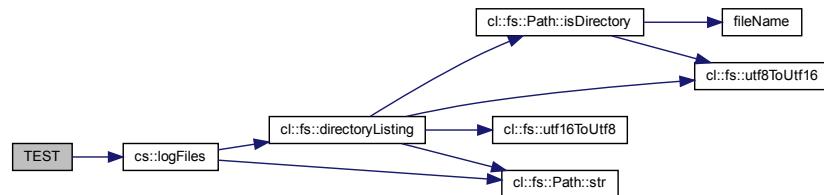


7.37.1.3 TEST() [3/3]

```
TEST (
    logFiles ,
    shouldNotFindGarbage )
```

Definition at line 40 of file log_files_test.cpp.

Here is the call graph for this function:

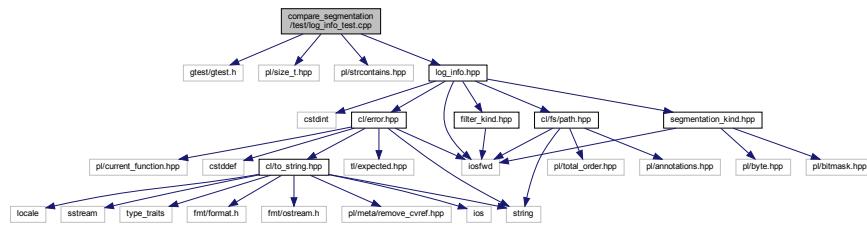


7.38 compare_segmentation/test/log_info_test.cpp File Reference

```
#include "gtest/gtest.h"
#include <pl/size_t.hpp>
#include <pl/strcontains.hpp>
```

```
#include "log_info.hpp"
```

Include dependency graph for log_info_test.cpp:



Functions

- [TEST](#) (LogInfo, shouldWork)
- [TEST](#) (LogInfo, shouldWork2)
- [TEST](#) (LogInfo, shouldWork3)
- [TEST](#) (LogInfo, shouldWork4)
- [TEST](#) (LogInfo, shouldWork5)
- [TEST](#) (LogInfo, shouldWork6)
- [TEST](#) (LogInfo, shouldWork7)
- [TEST](#) (LogInfo, shouldWork8)
- [TEST](#) (LogInfo, shouldWork9)
- [TEST](#) (LogInfo, shouldWorkWithPathOldPath)
- [TEST](#) (LogInfo, shouldWorkWithPathOldPath2)
- [TEST](#) (LogInfo, shouldResultInErrorIfLogFileWithPathIsTooShort)
- [TEST](#) (LogInfo, shouldFailIfSkipWindowIsInvalid)
- [TEST](#) (LogInfo, shouldFailIfDeleteTooClosesIsInvalid)
- [TEST](#) (LogInfo, shouldFailIfDeleteTooLowVarianceIsInvalid)
- [TEST](#) (LogInfo, shouldFailIfSegmentationKindIsInvalid)
- [TEST](#) (LogInfo, shouldFailIfWindowSizeIsInvalid)
- [TEST](#) (LogInfo, shouldFailIfFilterIsInvalid)
- [TEST](#) (LogInfo, shouldCreateUninitializedObjectWhenDefaultConstructorIsCalled)

7.38.1 Function Documentation

7.38.1.1 TEST() [1/19]

```
TEST (
    LogInfo ,
    shouldCreateUninitializedObjectWhenDefaultConstructorIsCalled )
```

Definition at line 388 of file log_info_test.cpp.

7.38.1.2 TEST() [2/19]

```
TEST (
    LogInfo ,
    shouldFailIfDeleteTooCloseIsInvalid )
```

Definition at line 341 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.3 TEST() [3/19]

```
TEST (
    LogInfo ,
    shouldFailIfDeleteTooLowVarianceIsInvalid )
```

Definition at line 350 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.4 TEST() [4/19]

```
TEST (
    LogInfo ,
    shouldFailIfFilterIsInvalid )
```

Definition at line 379 of file log_info_test.cpp.

Here is the call graph for this function:

**7.38.1.5 TEST() [5/19]**

```
TEST (
    LogInfo ,
    shouldFailIfSegmentationKindIsInvalid )
```

Definition at line 359 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.6 TEST() [6/19]

```
TEST (
    LogInfo ,
    shouldFailIfSkipWindowIsValid )
```

Definition at line 332 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.7 TEST() [7/19]

```
TEST (
    LogInfo ,
    shouldFailIfWindowSizeIsValid )
```

Definition at line 368 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.8 TEST() [8/19]

```
TEST (
    LogInfo ,
    shouldResultInErrorIfLogFilePathIsTooShort )
```

Definition at line 325 of file log_info_test.cpp.

Here is the call graph for this function:

**7.38.1.9 TEST() [9/19]**

```
TEST (
    LogInfo ,
    shouldWork )
```

Definition at line 8 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.10 TEST() [10/19]

```
TEST (
    LogInfo ,
    shouldWork2 )
```

Definition at line 37 of file log_info_test.cpp.

Here is the call graph for this function:

**7.38.1.11 TEST() [11/19]**

```
TEST (
    LogInfo ,
    shouldWork3 )
```

Definition at line 66 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.12 TEST() [12/19]

```
TEST (
    LogInfo ,
    shouldWork4 )
```

Definition at line 95 of file log_info_test.cpp.

Here is the call graph for this function:

**7.38.1.13 TEST() [13/19]**

```
TEST (
    LogInfo ,
    shouldWork5 )
```

Definition at line 124 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.14 TEST() [14/19]

```
TEST (
    LogInfo ,
    shouldWork6 )
```

Definition at line 153 of file log_info_test.cpp.

Here is the call graph for this function:

**7.38.1.15 TEST() [15/19]**

```
TEST (
    LogInfo ,
    shouldWork7 )
```

Definition at line 182 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.16 TEST() [16/19]

```
TEST (
    LogInfo ,
    shouldWork8 )
```

Definition at line 211 of file log_info_test.cpp.

Here is the call graph for this function:

**7.38.1.17 TEST() [17/19]**

```
TEST (
    LogInfo ,
    shouldWork9 )
```

Definition at line 240 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.18 TEST() [18/19]

```
TEST (
    LogInfo ,
    shouldWorkWithOldPath )
```

Definition at line 269 of file log_info_test.cpp.

Here is the call graph for this function:



7.38.1.19 TEST() [19/19]

```
TEST (
    LogInfo ,
    shouldWorkWithOldPath2 )
```

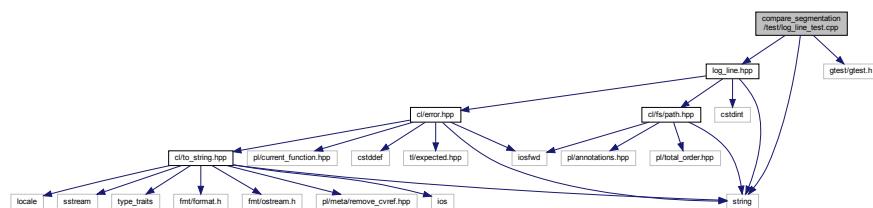
Definition at line 297 of file log_info_test.cpp.

Here is the call graph for this function:



7.39 compare_segmentation/test/log_line_test.cpp File Reference

```
#include <string>
#include "gtest/gtest.h"
#include "log_line.hpp"
Include dependency graph for log_line_test.cpp:
```



Functions

- [TEST](#) (LogLine, shouldWorkWithPreprocessedLine)
- [TEST](#) (LogLine, shouldWorkWithOldLine)
- [TEST](#) (LogLine, shouldNotMatchGarbage)
- [TEST](#) (LogLine, shouldNotParseGarbageSensor)

7.39.1 Function Documentation

7.39.1.1 TEST() [1/4]

```
TEST (
    LogLine ,
    shouldNotMatchGarbage )
```

Definition at line 41 of file log_line_test.cpp.

Here is the call graph for this function:



7.39.1.2 TEST() [2/4]

```
TEST (
    LogLine ,
    shouldNotParseGarbageSensor )
```

Definition at line 48 of file log_line_test.cpp.

Here is the call graph for this function:



7.39.1.3 TEST() [3/4]

```
TEST ( LogLine , shouldWorkWithOldLine )
```

Definition at line 25 of file log_line_test.cpp.

Here is the call graph for this function:



7.39.1.4 TEST() [4/4]

```
TEST ( LogLine , shouldWorkWithPreprocessedLine )
```

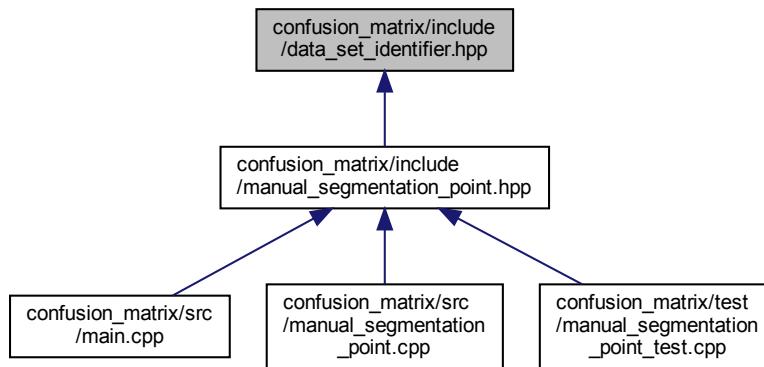
Definition at line 9 of file log_line_test.cpp.

Here is the call graph for this function:



7.40 confusion_matrix/include/data_set_identifier.hpp File Reference

This graph shows which files directly or indirectly include this file:



Namespaces

- cm

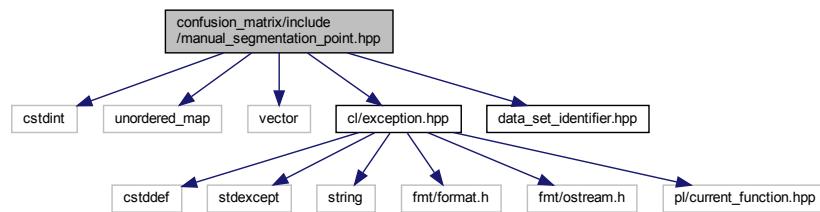
Enumerations

- enum cm::DataSetIdentifier {
 cm::DataSetIdentifier::Felix_11_17_39, cm::DataSetIdentifier::Felix_12_50_00, cm::DataSetIdentifier::Felix_13_00_09,
 cm::DataSetIdentifier::Mike_14_07_33,
 cm::DataSetIdentifier::Mike_14_14_32, cm::DataSetIdentifier::Mike_14_20_28, cm::DataSetIdentifier::Marsi_14_59_59,
 cm::DataSetIdentifier::Marsi_15_13_22,
 cm::DataSetIdentifier::Marsi_15_31_36, cm::DataSetIdentifier::Jan_1, cm::DataSetIdentifier::Jan_2,
 cm::DataSetIdentifier::Jan_3,
 cm::DataSetIdentifier::Andre_1, cm::DataSetIdentifier::Andre_2, cm::DataSetIdentifier::Andre_3, cm::DataSetIdentifier::Andre_4,
 cm::DataSetIdentifier::Andre_Squats_2, cm::DataSetIdentifier::Lucas_1, cm::DataSetIdentifier::Lucas_2,
 cm::DataSetIdentifier::Lucas_3
 }

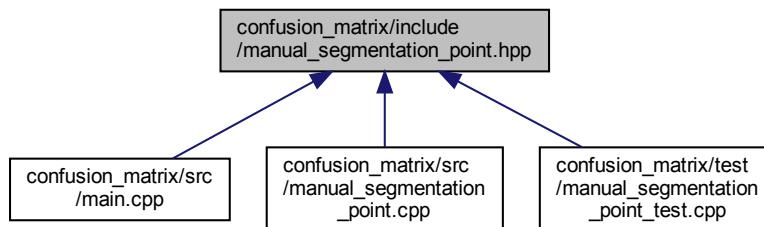
7.41 confusion_matrix/include/manual_segmentation_point.hpp File Reference

```
#include <cstdint>
#include <unordered_map>
#include <vector>
#include <c1/exception.hpp>
```

```
#include "data_set_identifier.hpp"
Include dependency graph for manual_segmentation_point.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [cm::ManualSegmentationPoint](#)
Type used to represent a manual segmentation point.

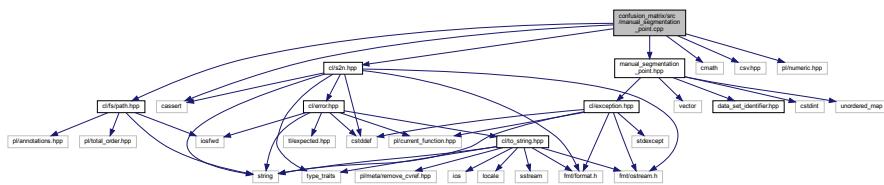
Namespaces

- [cm](#)

7.42 confusion_matrix/src/manual_segmentation_point.cpp File Reference

```
#include <cassert>
#include <cmath>
#include <csv.hpp>
#include <pl/numeric.hpp>
#include "cl/fs/path.hpp"
#include "cl/s2n.hpp"
```

```
#include "manual_segmentation_point.hpp"
Include dependency graph for manual_segmentation_point.cpp:
```



Namespaces

- cm

Macros

- #define DSI DataSetIdentifier

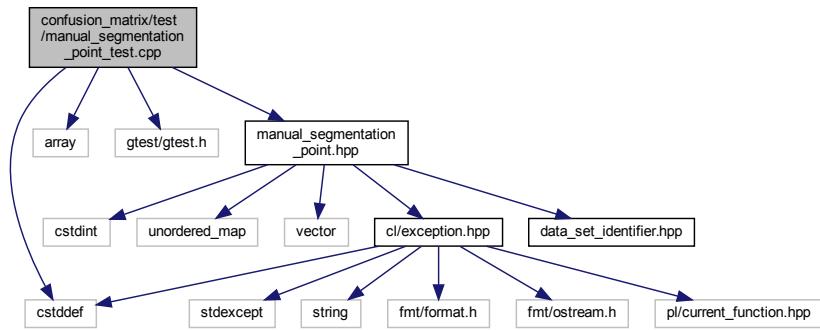
7.42.1 Macro Definition Documentation

7.42.1.1 DSI

```
#define DSI DataSetIdentifier
```

7.43 confusion_matrix/test/manual_segmentation_point_test.cpp File Reference

```
#include <cstddef>
#include <array>
#include "gtest/gtest.h"
#include "manual_segmentation_point.hpp"
Include dependency graph for manual_segmentation_point_test.cpp:
```



Functions

- [TEST \(ManualSegmentationPoint, shouldConstruct\)](#)
- [TEST \(ManualSegmentationPoint, shouldThrowWhenConstructingWithInvalidMinute\)](#)
- [TEST \(ManualSegmentationPoint, shouldThrowWhenConstructingWithInvalidSecond\)](#)
- [TEST \(ManualSegmentationPoint, shouldThrowWhenConstructingWithInvalidFrame\)](#)
- [TEST \(ManualSegmentationPoint, shouldConvertToMilliseconds\)](#)
- [TEST \(ManualSegmentationPoint, shouldConvertHourToMilliseconds\)](#)
- [TEST \(ManualSegmentationPoint, shouldConvertMinuteToMilliseconds\)](#)
- [TEST \(ManualSegmentationPoint, shouldConvertSecondToMilliseconds\)](#)
- [TEST \(ManualSegmentationPoint, shouldConvertFramesToMilliseconds\)](#)

7.43.1 Function Documentation

7.43.1.1 TEST() [1/9]

```
TEST (
    ManualSegmentationPoint ,
    shouldConstruct )
```

Definition at line 9 of file manual_segmentation_point_test.cpp.

7.43.1.2 TEST() [2/9]

```
TEST (
    ManualSegmentationPoint ,
    shouldConvertFramesToMilliseconds )
```

Definition at line 79 of file manual_segmentation_point_test.cpp.

7.43.1.3 TEST() [3/9]

```
TEST (
    ManualSegmentationPoint ,
    shouldConvertHourToMilliseconds )
```

Definition at line 61 of file manual_segmentation_point_test.cpp.

7.43.1.4 TEST() [4/9]

```
TEST (
    ManualSegmentationPoint ,
    shouldConvertMinuteToMilliseconds )
```

Definition at line 67 of file manual_segmentation_point_test.cpp.

7.43.1.5 TEST() [5/9]

```
TEST (
    ManualSegmentationPoint ,
    shouldConvertSecondToMilliseconds )
```

Definition at line 73 of file manual_segmentation_point_test.cpp.

7.43.1.6 TEST() [6/9]

```
TEST (
    ManualSegmentationPoint ,
    shouldConvertToMilliseconds )
```

Definition at line 55 of file manual_segmentation_point_test.cpp.

7.43.1.7 TEST() [7/9]

```
TEST (
    ManualSegmentationPoint ,
    shouldThrowWhenConstructingWithInvalidFrame )
```

Definition at line 43 of file manual_segmentation_point_test.cpp.

7.43.1.8 TEST() [8/9]

```
TEST (
    ManualSegmentationPoint ,
    shouldThrowWhenConstructingWithInvalidMinute )
```

Definition at line 19 of file manual_segmentation_point_test.cpp.

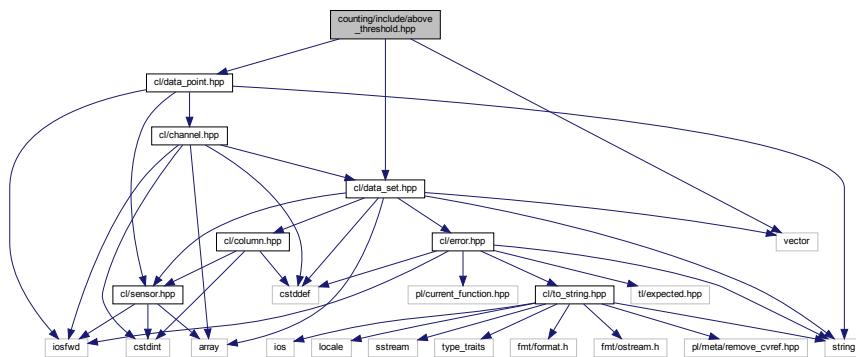
7.43.1.9 TEST() [9/9]

```
TEST (
    ManualSegmentationPoint ,
    shouldThrowWhenConstructingWithInvalidSecond )
```

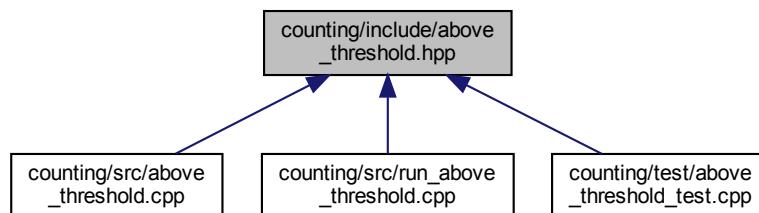
Definition at line 31 of file manual_segmentation_point_test.cpp.

7.44 counting/include/above_threshold.hpp File Reference

```
#include <vector>
#include "cl/data_point.hpp"
#include "cl/data_set.hpp"
Include dependency graph for above_threshold.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

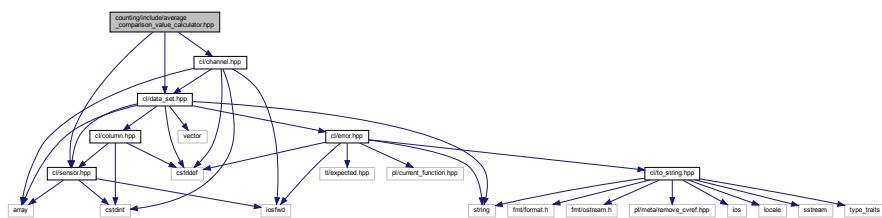
- ctg

Functions

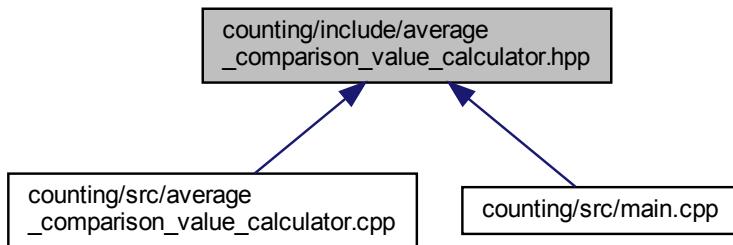
- std::vector< cl::DataPoint > ctg::aboveThreshold (const cl::DataSet &dataSet, long double accelerometerThreshold, long double gyroscopeThreshold)

7.45 counting/include/average_comparison_value_calculator.hpp File Reference

```
#include "cl/channel.hpp"
#include "cl/data_set.hpp"
#include "cl/sensor.hpp"
Include dependency graph for average_comparison_value_calculator.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

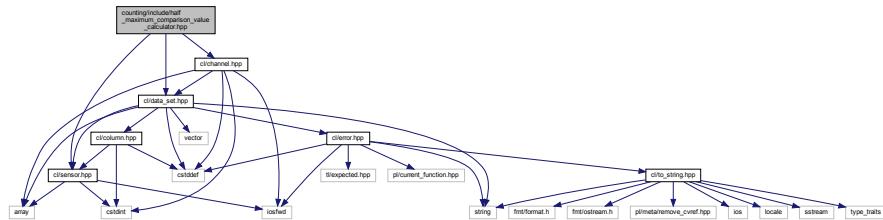
- `ctg`

Functions

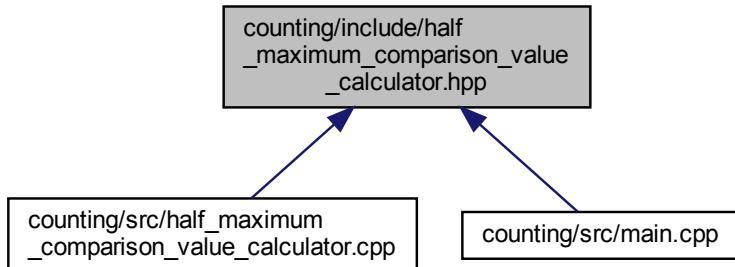
- long double `ctg::averageComparisonValueCalculator` (`cl::Sensor sensor`, `cl::Channel channel`, `const cl::DataSet &dataSet)`

7.46 counting/include/half_maximum_comparison_value_calculator.hpp File Reference

```
#include "cl/channel.hpp"
#include "cl/data_set.hpp"
#include "cl/sensor.hpp"
Include dependency graph for half_maximum_comparison_value_calculator.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

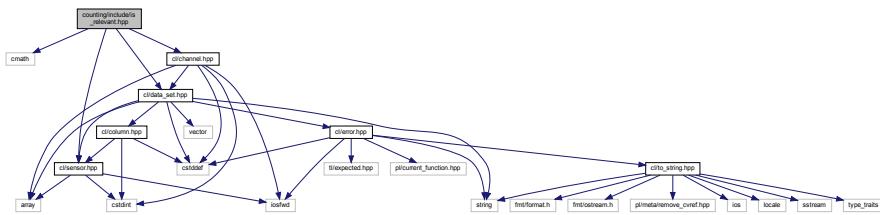
- ctg

Functions

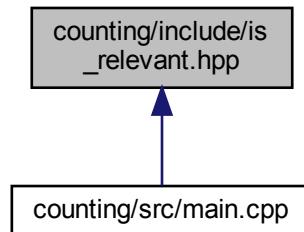
- long double `ctg::halfMaximumComparisonValueCalculator` (`cl::Sensor sensor, cl::Channel channel, const cl::DataSet &dataSet`)

7.47 counting/include/is_relevant.hpp File Reference

```
#include <cmath>
#include "cl/channel.hpp"
#include "cl/data_set.hpp"
#include "cl/sensor.hpp"
Include dependency graph for is_relevant.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

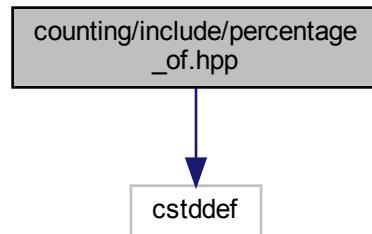
- `ctg`

Functions

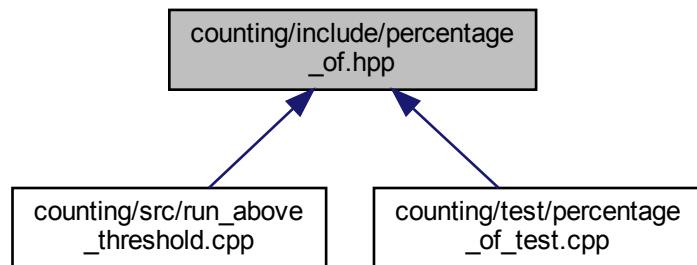
- template<typename ComparisonValueCalculator >
`bool ctg::isRelevant (cl::Sensor sensor, cl::Channel channel, const cl::DataSet &dataSet, ComparisonValueCalculator comparisonValueCalculator)`

7.48 counting/include/percentage_of.hpp File Reference

```
#include <cstdint>
Include dependency graph for percentage_of.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

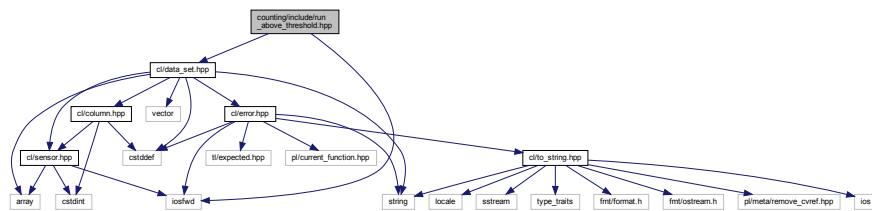
- [ctg](#)

Functions

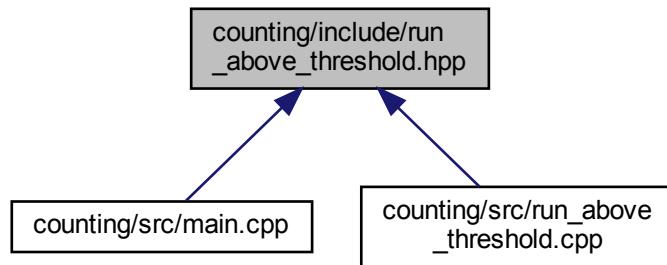
- `constexpr long double ctg::percentageOf (std::size_t amount, std::size_t totalCount) noexcept`

7.49 counting/include/run_above_threshold.hpp File Reference

```
#include <iostream>
#include "cl/data_set.hpp"
Include dependency graph for run_above_threshold.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

- `ctg`

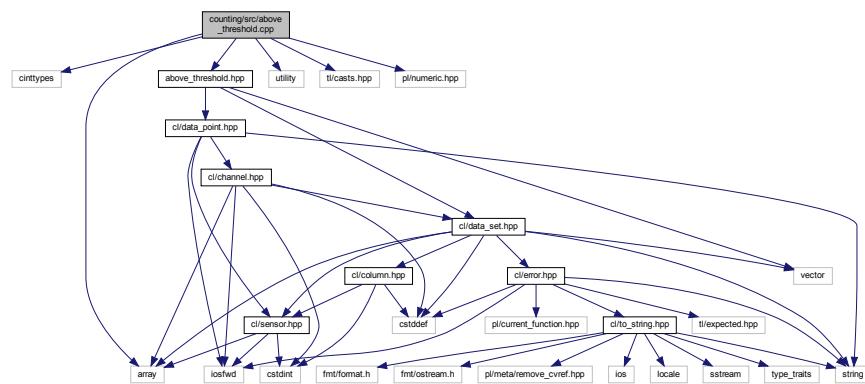
Functions

- void `ctg::runAboveThreshold` (`std::ostream &aboveThresholdLogFileStream, const cl::DataSet &dataSet)`

7.50 counting/src/above_threshold.cpp File Reference

```
#include <cinttypes>
#include <array>
#include <utility>
#include <tl/casts.hpp>
#include <pl/numeric.hpp>
```

```
#include "above_threshold.hpp"
Include dependency graph for above_threshold.cpp:
```



Namespaces

- `ctg`

Macros

- `#define CL_CHANNEL_X(enm, v, accessor) {accessor, cl::Channel::enm},`

Functions

- `std::vector< cl::DataPoint > ctg::aboveThreshold (const cl::DataSet &dataSet, long double accelerometerThreshold, long double gyroscopeThreshold)`

7.50.1 Macro Definition Documentation

7.50.1.1 CL_CHANNEL_X

```
#define CL_CHANNEL_X(
    enm,
    v,
    accessor ) {accessor, cl::Channel::enm},
```

7.50.2 Variable Documentation

7.50.2.1 channel

```
cl::Channel channel
```

Definition at line 18 of file above_threshold.cpp.

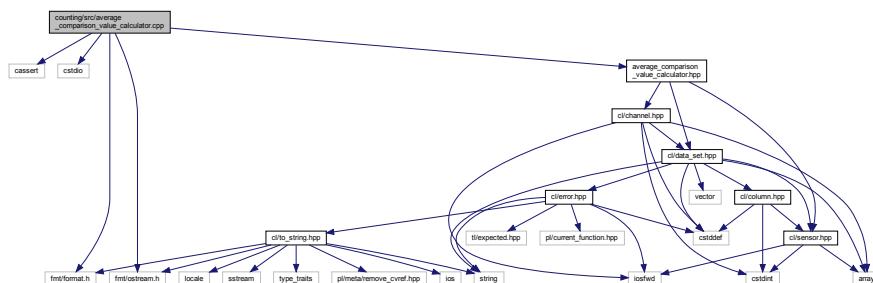
7.50.2.2 channelAccessor

```
cl::DataSet::ChannelAccessor channelAccessor
```

Definition at line 17 of file above_threshold.cpp.

7.51 counting/src/average_comparison_value_calculator.cpp File Reference

```
#include <cassert>
#include <cstdio>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include "average_comparison_value_calculator.hpp"
Include dependency graph for average_comparison_value_calculator.cpp:
```



Namespaces

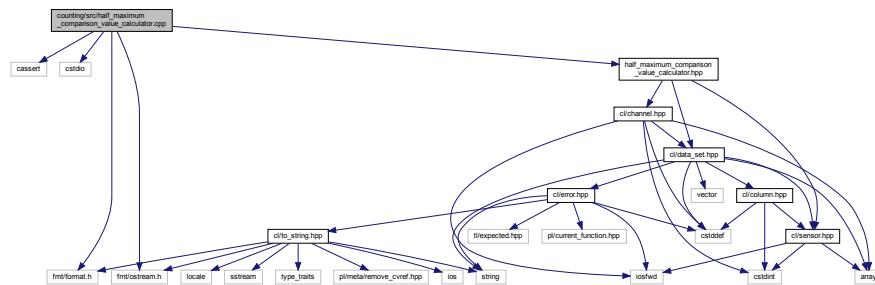
- `ctg`

Functions

- long double `ctg::averageComparisonValueCalculator (cl::Sensor sensor, cl::Channel channel, const cl::DataSet &dataSet)`

7.52 counting/src/half_maximum_comparison_value_calculator.cpp File Reference

```
#include <cassert>
#include <cstdio>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include "half_maximum_comparison_value_calculator.hpp"
Include dependency graph for half_maximum_comparison_value_calculator.cpp:
```



Namespaces

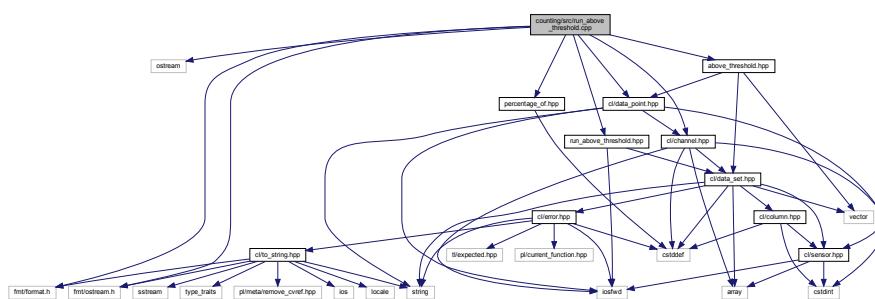
- ctg

Functions

- long double `ctg::halfMaximumComparisonValueCalculator` (`cl::Sensor sensor, cl::Channel channel, const cl::DataSet &dataSet`)

7.53 counting/src/run_above_threshold.cpp File Reference

```
#include <iostream>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include "cl/channel.hpp"
#include "cl/data_point.hpp"
#include "above_threshold.hpp"
#include "percentage_of.hpp"
#include "run_above_threshold.hpp"
Include dependency graph for run_above_threshold.cpp:
```



Namespaces

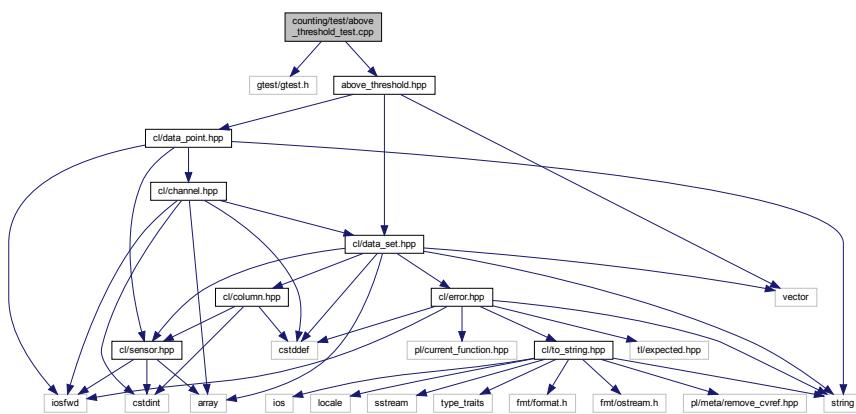
- `ctg`

Functions

- void `ctg::runAboveThreshold` (`std::ostream &aboveThresholdLogFileStream, const cl::DataSet &dataSet)`

7.54 counting/test/above_threshold_test.cpp File Reference

```
#include "gtest/gtest.h"
#include "above_threshold.hpp"
Include dependency graph for above_threshold_test.cpp:
```



Macros

- `#define EXPECT_LONG_DOUBLE_EQ(a, b) EXPECT_DOUBLE_EQ(static_cast<double>(a), static_cast<double>(b))`

Functions

- `TEST` (`aboveThreshold, shouldFindDataPointsIfThereAreAny`)

7.54.1 Macro Definition Documentation

7.54.1.1 EXPECT_LONG_DOUBLE_EQ

```
#define EXPECT_LONG_DOUBLE_EQ( a, b ) EXPECT_DOUBLE_EQ( static_cast<double>(a), static_cast<double>(b) )
```

Definition at line 6 of file above_threshold_test.cpp.

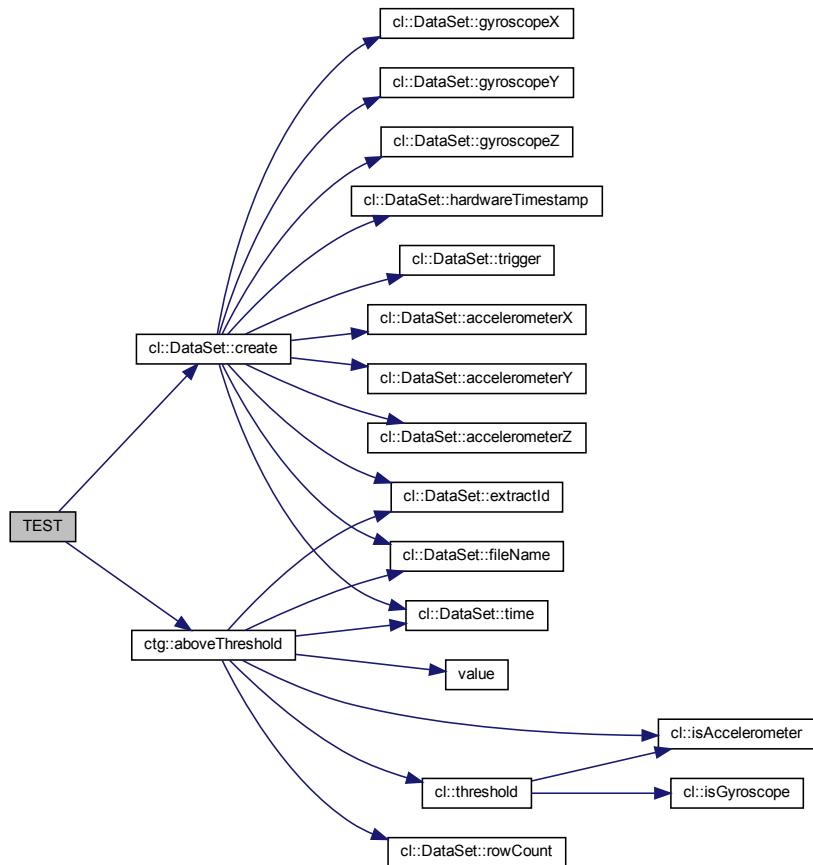
7.54.2 Function Documentation

7.54.2.1 TEST()

```
TEST( aboveThreshold , shouldFindDataPointsIfThereAreAny )
```

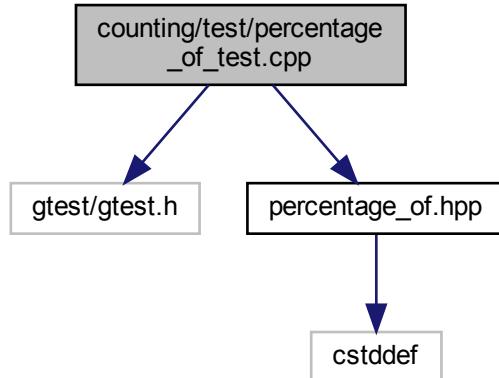
Definition at line 10 of file above_threshold_test.cpp.

Here is the call graph for this function:



7.55 counting/test/percentage_of_test.cpp File Reference

```
#include "gtest/gtest.h"
#include "percentage_of.hpp"
Include dependency graph for percentage_of_test.cpp:
```



Macros

- `#define EXPECT_LONG_DOUBLE_EQ(a, b) EXPECT_DOUBLE_EQ(static_cast<double>(a), static_cast<double>(b))`

Functions

- `TEST(percentageOf, shouldWork)`

7.55.1 Macro Definition Documentation

7.55.1.1 EXPECT_LONG_DOUBLE_EQ

```
#define EXPECT_LONG_DOUBLE_EQ(
    a,
    b ) EXPECT_DOUBLE_EQ(static_cast<double>(a), static_cast<double>(b))
```

Definition at line 6 of file percentage_of_test.cpp.

7.55.2 Function Documentation

7.55.2.1 TEST()

```
TEST (
    percentageOf ,
    shouldWork )
```

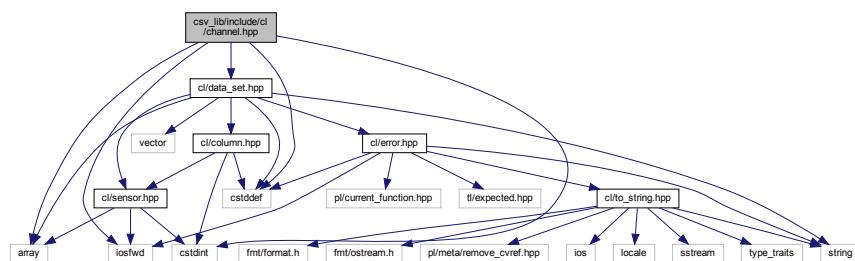
Definition at line 10 of file percentage_of_test.cpp.

Here is the call graph for this function:

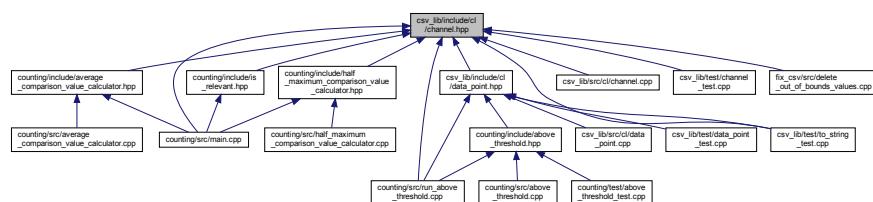


7.56 csv_lib/include/cl/channel.hpp File Reference

```
#include <cstdint>
#include <array>
#include <iostream>
#include "cl/data_set.hpp"
Include dependency graph for channel.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct `cl::data_set_accessor< Chan >`

Namespaces

- `cl`

Macros

- `#define CL_CHANNEL`
- `#define CL_CHANNEL_X(enumerator, value, dataSetAccessor) enumerator = value,`
- `#define CL_CHANNEL_X(enumerator, value, dataSetAccessor) +1`
- `#define CL_CHANNEL_X(enm, v, a) ::cl::Channel::enm,`
- `#define CL_CHANNEL_X(enumerator, value, dataSetAccessor)`

Enumerations

- enum `cl::Channel : std::uint64_t { CL_CHANNEL, CL_CHANNEL }`

Functions

- `DataSet::ChannelAccessor cl::dataSetAccessor (Channel channel)`
- `std::ostream & cl::operator<< (std::ostream &os, Channel channel)`
- `bool cl::isAccelerometer (Channel channel)`
- `bool cl::isGyroscope (Channel channel)`
- `long double cl::threshold (Channel channel)`

Variables

- `constexpr std::size_t cl::channelCount`
- `constexpr std::array< Channel, channelCount > cl::channels`
- `template<Channel Chan>`
`constexpr CL_CHANNEL DataSet::ChannelAccessor cl::data_set_accessor_v = data_set_accessor<Chan>::f`
- `constexpr long double cl::accelerometerThreshold {1.99L}`
- `constexpr long double cl::gyroscopeThreshold {1999.99L}`

7.56.1 Macro Definition Documentation

7.56.1.1 CL_CHANNEL

```
#define CL_CHANNEL
```

Value:

```
CL_CHANNEL_X(AccelerometerX, 1, &::cl::DataSet::accelerometerX) \
CL_CHANNEL_X(AccelerometerY, 2, &::cl::DataSet::accelerometerY) \
CL_CHANNEL_X(AccelerometerZ, 3, &::cl::DataSet::accelerometerZ) \
CL_CHANNEL_X(GyroscopeX, 4, &::cl::DataSet::gyroscopeX) \
CL_CHANNEL_X(GyroscopeY, 5, &::cl::DataSet::gyroscopeY) \
CL_CHANNEL_X(GyroscopeZ, 6, &::cl::DataSet::gyroscopeZ)
```

Definition at line 11 of file channel.hpp.

7.56.1.2 CL_CHANNEL_X [1/4]

```
#define CL_CHANNEL_X(
    enm,
    v,
    a ) ::cl::Channel::enm,
```

Definition at line 41 of file channel.hpp.

7.56.1.3 CL_CHANNEL_X [2/4]

```
#define CL_CHANNEL_X(
    enumerator,
    value,
    dataSetAccessor ) enumerator = value,
```

Definition at line 41 of file channel.hpp.

7.56.1.4 CL_CHANNEL_X [3/4]

```
#define CL_CHANNEL_X(
    enumerator,
    value,
    dataSetAccessor ) +1
```

Definition at line 41 of file channel.hpp.

7.56.1.5 CL_CHANNEL_X [4/4]

```
#define CL_CHANNEL_X(enumerator, value, dataSetAccessor )
```

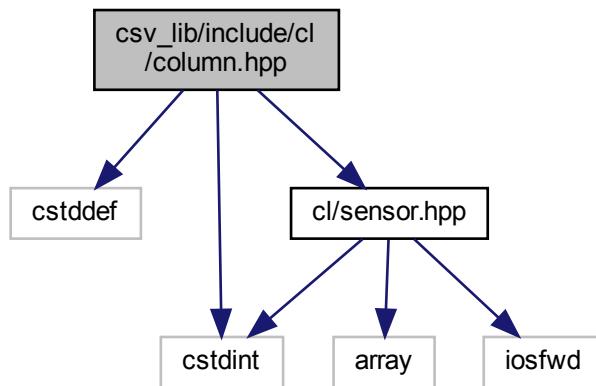
Value:

```
template<>
struct data_set_accessor<Channel::enumerator> {
    static constexpr ::cl::DataSet::ChannelAccessor f = dataSetAccessor; \
};
```

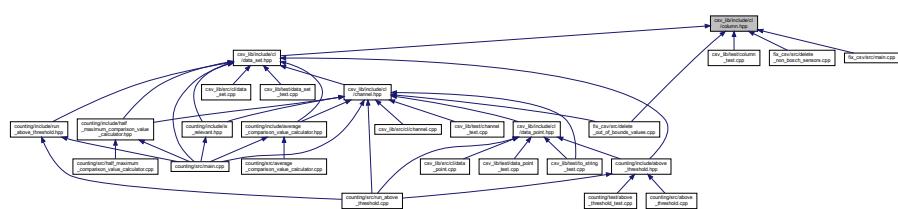
Definition at line 41 of file channel.hpp.

7.57 csv_lib/include/cl/column.hpp File Reference

```
#include <cstdint>
#include "cl/sensor.hpp"
Include dependency graph for column.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- struct `cl::col_traits< Col >`

Namespaces

- `cl`

Macros

- `#define CL_SPECIALIZE_COL_TRAITS(column, columnType)`

Typedefs

- template<Column Col>
using `cl::column_type` = typename `col_traits< Col >::type`

Enumerations

- enum `cl::Column` : `std::size_t` {
`cl::Column::Time, cl::Column::HardwareTimestamp, cl::Column::ExtractId, cl::Column::Trigger,`
`cl::Column::AccelerometerX, cl::Column::AccelerometerY, cl::Column::AccelerometerZ, cl::Column::GyroscopeX,`
`cl::Column::GyroscopeY, cl::Column::GyroscopeZ, cl::Column::SamplingRate }`

Functions

- `cl::CL_SPECIALIZE_COL_TRAITS (Column::Time, long double)`
- `cl::CL_SPECIALIZE_COL_TRAITS (Column::HardwareTimestamp, std::uint64_t)`
- `cl::CL_SPECIALIZE_COL_TRAITS (Column::ExtractId, Sensor)`
- `cl::CL_SPECIALIZE_COL_TRAITS (Column::Trigger, std::uint64_t)`
- `cl::CL_SPECIALIZE_COL_TRAITS (Column::AccelerometerX, long double)`
- `cl::CL_SPECIALIZE_COL_TRAITS (Column::AccelerometerY, long double)`
- `cl::CL_SPECIALIZE_COL_TRAITS (Column::AccelerometerZ, long double)`
- `cl::CL_SPECIALIZE_COL_TRAITS (Column::GyroscopeX, long double)`
- `cl::CL_SPECIALIZE_COL_TRAITS (Column::GyroscopeY, long double)`
- `cl::CL_SPECIALIZE_COL_TRAITS (Column::GyroscopeZ, long double)`
- `cl::CL_SPECIALIZE_COL_TRAITS (Column::SamplingRate, std::uint64_t)`

Variables

- template<Column Col>
constexpr `std::size_t cl::column_index` = `col_traits<Col>::index`

7.57.1 Macro Definition Documentation

7.57.1.1 CL_SPECIALIZE_COL_TRAITS

```
#define CL_SPECIALIZE_COL_TRAITS(  
    column,  
    columnType )
```

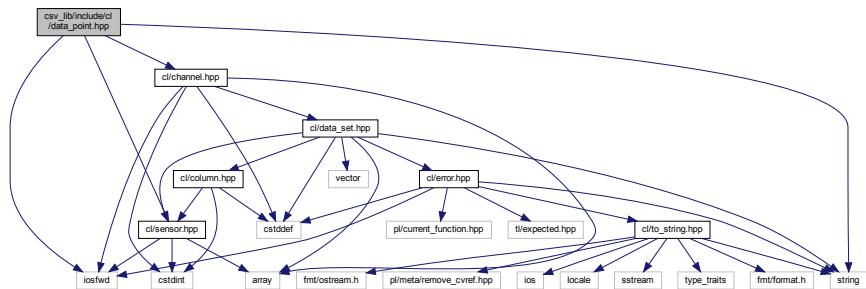
Value:

```
template<>
struct col_traits<column> {
    static constexpr std::size_t index = static_cast<std::size_t>(column);
    using type                      = columnType;
}
```

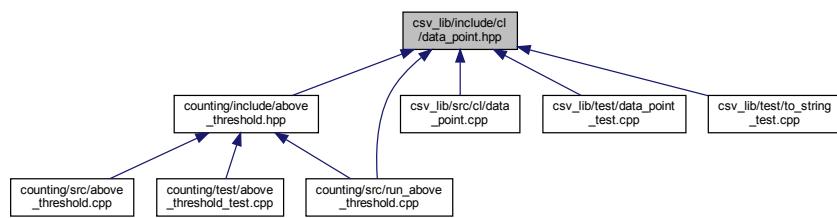
Definition at line 26 of file column.hpp.

7.58 csv_lib/include/cl/data_point.hpp File Reference

```
#include <iostream>
#include <string>
#include "cl/channel.hpp"
#include "cl/sensor.hpp"
Include dependency graph for data_point.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

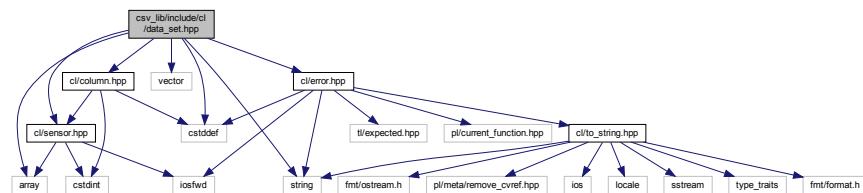
- class cl::DataPoint

Namespaces

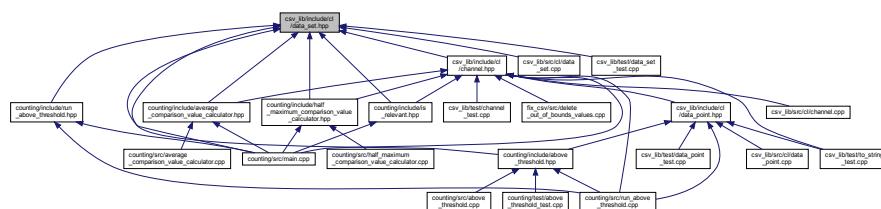
- [cl](#)

7.59 csv_lib/include/cl/data_set.hpp File Reference

```
#include <cstddef>
#include <array>
#include <string>
#include <vector>
#include "cl/column.hpp"
#include "cl/error.hpp"
#include "cl/sensor.hpp"
Include dependency graph for data_set.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

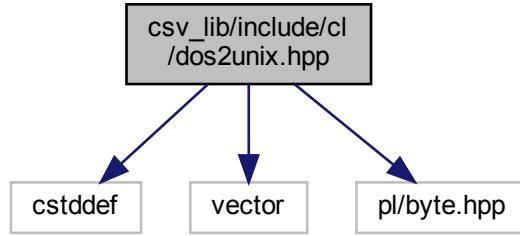
- class [cl::DataSet](#)

Namespaces

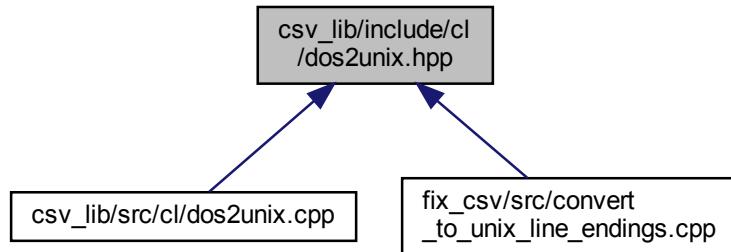
- [cl](#)

7.60 csv_lib/include/cl/dos2unix.hpp File Reference

```
#include <cstddef>
#include <vector>
#include <pl/byte.hpp>
Include dependency graph for dos2unix.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

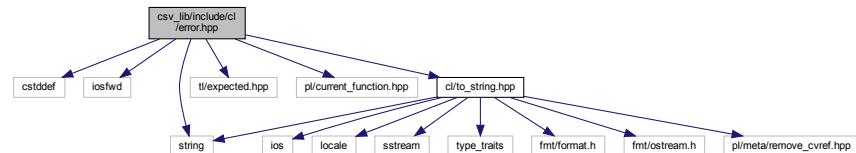
- `cl`

Functions

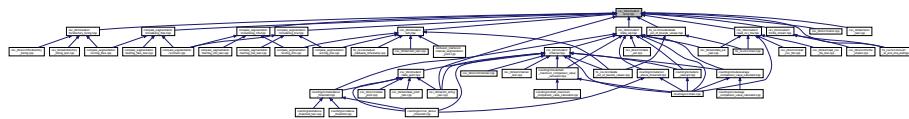
- `std::vector< pl::byte > cl::dos2unix (const void *p, std::size_t size)`
Converts DOS / Microsoft Windows line endings to UNIX line endings.

7.61 csv_lib/include/cl/error.hpp File Reference

```
#include <cstddef>
#include <iostream>
#include <string>
#include <t1/expected.hpp>
#include <pl/current_function.hpp>
#include "cl/to_string.hpp"
Include dependency graph for error.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [cl::Error](#)

Namespaces

- [cl](#)

Macros

- `#define CL_ERROR_KIND`
- `#define CL_ERROR_KIND_X(kind) kind,`
- `#define CL_UNEXPECTED(kind, message)`

Typedefs

- template<typename Ty >
using [cl::Expected](#) = t1::expected< Ty, Error >

7.61.1 Macro Definition Documentation

7.61.1.1 CL_ERROR_KIND

```
#define CL_ERROR_KIND
```

Value:

```
CL_ERROR_KIND_X(Filesystem) \
CL_ERROR_KIND_X(InvalidArgumentException) \
CL_ERROR_KIND_X(OutOfRange) \
CL_ERROR_KIND_X(Parsing) \
CL_ERROR_KIND_X(Logic)
```

Definition at line 14 of file error.hpp.

7.61.1.2 CL_ERROR_KIND_X

```
#define CL_ERROR_KIND_X( \
    kind ) kind,
```

Definition at line 26 of file error.hpp.

7.61.1.3 CL_UNEXPECTED

```
#define CL_UNEXPECTED( \
    kind, \
    message )
```

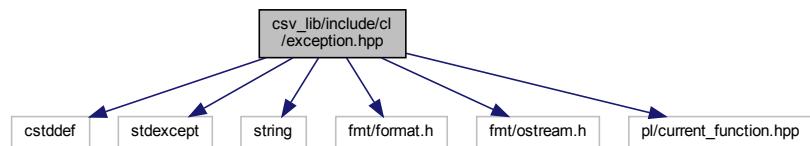
Value:

```
::tl::make_unexpected( \
    ::cl::Error{kind, __FILE__, PL_CURRENT_FUNCTION, __LINE__, message})
```

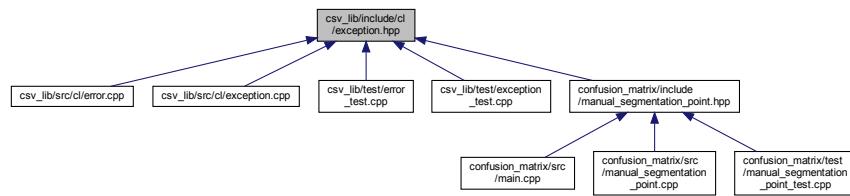
Definition at line 66 of file error.hpp.

7.62 csv_lib/include/cl/exception.hpp File Reference

```
#include <cstddef>
#include <stdexcept>
#include <string>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/current_function.hpp>
Include dependency graph for exception.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [cl::Exception](#)

Namespaces

- [cl](#)

Macros

- `#define CL_THROW(what_arg) throw ::cl::Exception { __FILE__, PL_CURRENT_FUNCTION, __LINE__ ← , what_arg }`
- `#define CL_THROW_FMT(fmt_str, ...) CL_THROW(::fmt::format(fmt_str, __VA_ARGS__))`

7.62.1 Macro Definition Documentation

7.62.1.1 CL_THROW

```
#define CL_THROW(
    what_arg ) throw ::cl::Exception { __FILE__, PL_CURRENT_FUNCTION, __LINE__ ← ,
    what_arg }
```

Definition at line 42 of file exception.hpp.

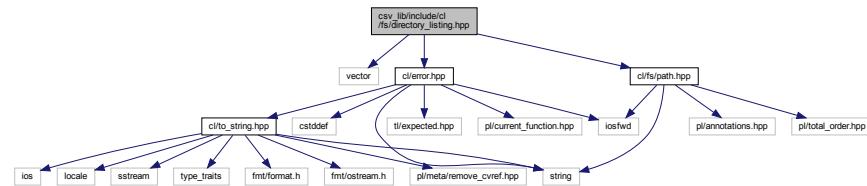
7.62.1.2 CL_THROW_FMT

```
#define CL_THROW_FMT(
    fmt_str,
    ... ) CL_THROW(::fmt::format(fmt_str, __VA_ARGS__))
```

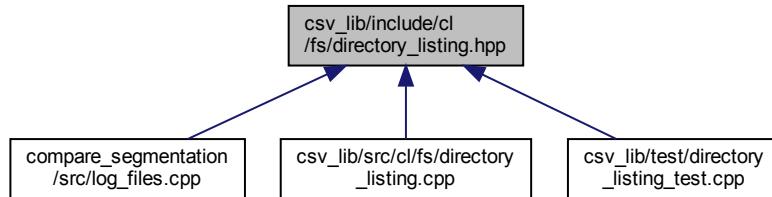
Definition at line 45 of file exception.hpp.

7.63 csv_lib/include/cl/fs/directory_listing.hpp File Reference

```
#include <vector>
#include <cl/error.hpp>
#include <cl/fs/path.hpp>
Include dependency graph for directory_listing.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

- `cl`
- `cl::fs`

Enumerations

- enum `cl::fs::DirectoryListingOption` { `cl::fs::DirectoryListingOption::None`, `cl::fs::DirectoryListingOption::ExcludeDotAndDotDot` }

Options for directoryListing.

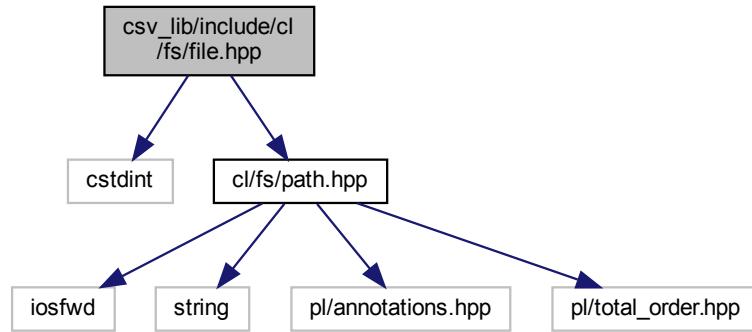
Functions

- `Expected< std::vector< Path > > cl::fs::directoryListing (const Path &directoryPath, DirectoryListingOption directoryListingOption=DirectoryListingOption::ExcludeDotAndDotDot)`

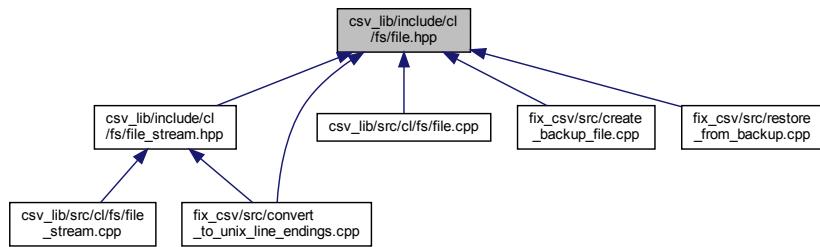
Creates a listing of the contents of a directory.

7.64 csv_lib/include/cl/fs/file.hpp File Reference

```
#include <cstdint>
#include "cl/fs/path.hpp"
Include dependency graph for file.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [cl::fs::File](#)

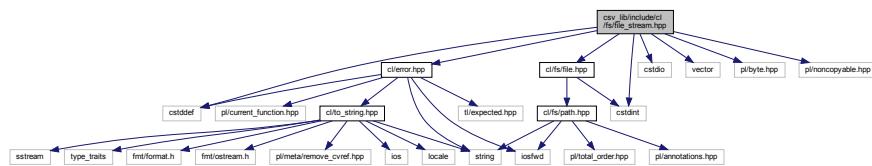
Represents a file.

Namespaces

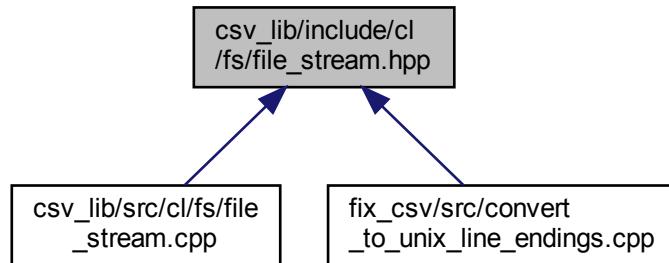
- [cl](#)
- [cl::fs](#)

7.65 csv_lib/include/cl/fs/file_stream.hpp File Reference

```
#include <cstddef>
#include <cstdint>
#include <cstdio>
#include <vector>
#include <pl/byte.hpp>
#include <pl/noncopyable.hpp>
#include "cl/error.hpp"
#include "cl/fs/file.hpp"
Include dependency graph for file_stream.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [cl::fs::FileStream](#)

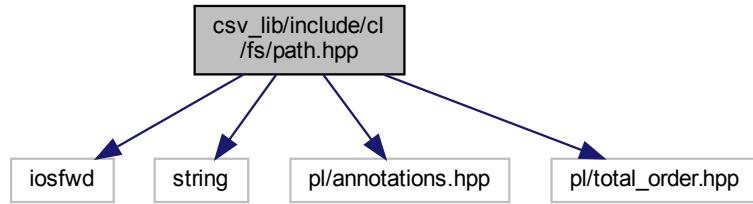
A binary file stream.

Namespaces

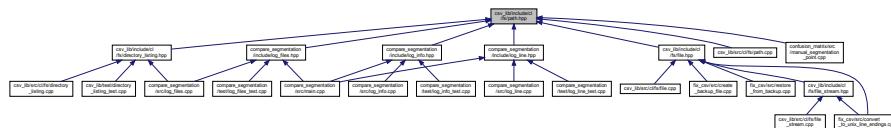
- [cl](#)
- [cl::fs](#)

7.66 csv_lib/include/cl/fs/path.hpp File Reference

```
#include <iostream>
#include <string>
#include <pl/annotations.hpp>
#include <pl/total_order.hpp>
Include dependency graph for path.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class `cl::fs::Path`

A filesystem path.

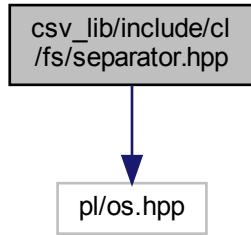
Namespaces

- cl
 - cl::fs

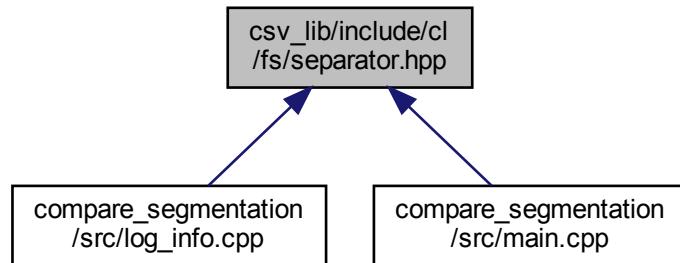
7.67 csv_lib/include/cl/fs/separator.hpp File Reference

```
#include <pl/os.hpp>
```

Include dependency graph for separator.hpp:



This graph shows which files directly or indirectly include this file:



Macros

- `#define CL_FS_SEPARATOR "\\\"`
The filesystem separator of the operating system.

7.67.1 Macro Definition Documentation

7.67.1.1 CL_FS_SEPARATOR

```
#define CL_FS_SEPARATOR "\\\"
```

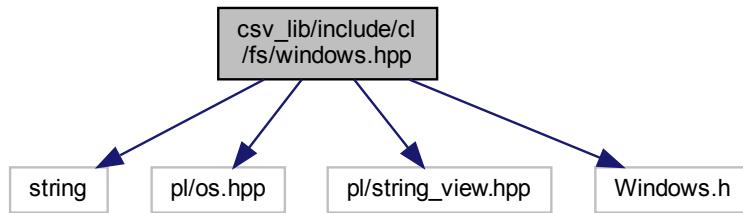
The filesystem separator of the operating system.

Definition at line 11 of file separator.hpp.

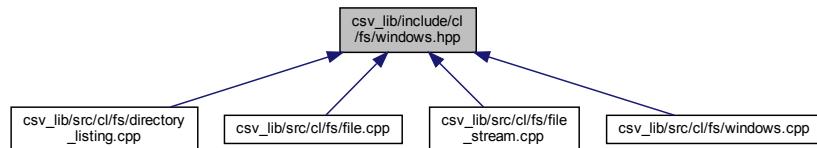
7.68 csv_lib/include/cl/fs/windows.hpp File Reference

Contains Microsoft Windows specific functions.

```
#include <string>
#include <pl/os.hpp>
#include <pl/string_view.hpp>
#include <Windows.h>
Include dependency graph for windows.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

- `cl`
- `cl::fs`

Functions

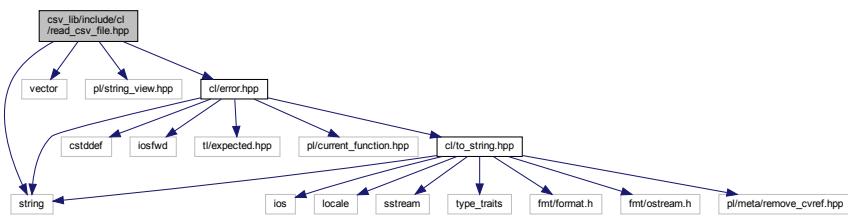
- `std::wstring cl::fs::utf8ToUtf16 (pl::string_view utf8)`
Converts a UTF-8 encoded string to a UTF-16 encoded wstring.
- `std::string cl::fs::utf16ToUtf8 (pl::wstring_view utf16)`
Converts a UTF-16 encoded wide character string to UTF-8 string.
- `std::wstring cl::fs::formatError (DWORD errorCode)`
Formats a WINAPI error code to a UTF-16 encoded wide character string.

7.68.1 Detailed Description

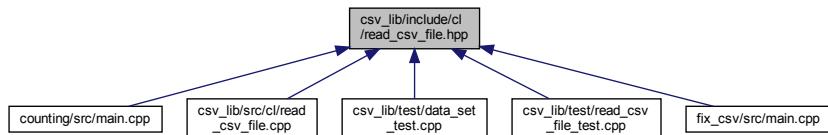
Contains Microsoft Windows specific functions.

7.69 csv_lib/include/cl/read_csv_file.hpp File Reference

```
#include <string>
#include <vector>
#include <pl/string_view.hpp>
#include "cl/error.hpp"
Include dependency graph for read_csv_file.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

- `cl`

Enumerations

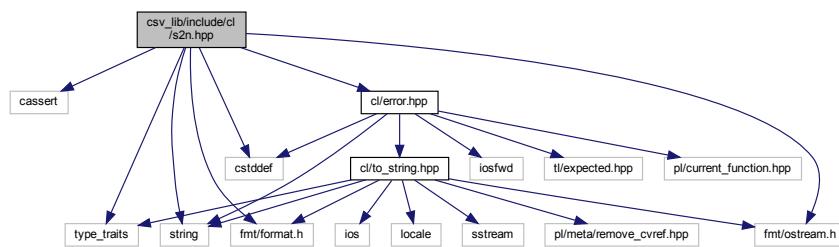
- enum `cl::CsvFileKind` { `cl::CsvFileKind::Raw`, `cl::CsvFileKind::Fixed` }

Functions

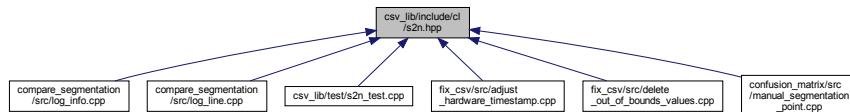
- `Expected< std::vector< std::vector< std::string > >> cl::readCsvFile(pl::string_view csvFilePath, std::vector< std::string > *columnNames=nullptr, CsvFileKind csvFileKind=CsvFileKind::Fixed) noexcept`

7.70 csv_lib/include/cl/s2n.hpp File Reference

```
#include <cassert>
#include <cstddef>
#include <string>
#include <type_traits>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include "cl/error.hpp"
Include dependency graph for s2n.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

- `cl`

Functions

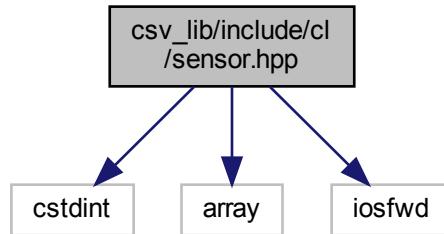
- template<typename Integer >
Expected< Integer > `cl::s2n` (const std::string &str, std::size_t *pos=nullptr, [[maybe_unused]] int base=10)

7.71 csv_lib/include/cl/sensor.hpp File Reference

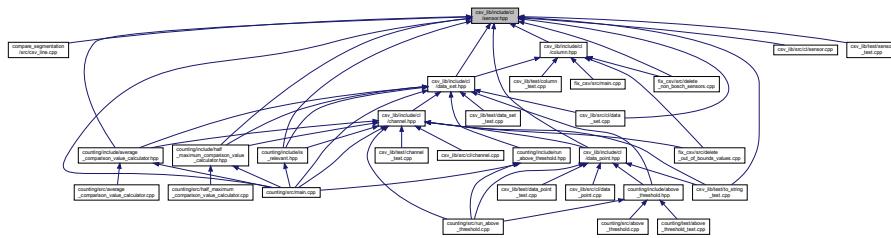
```
#include <cstdint>
#include <array>
```

```
#include <iostream>
```

Include dependency graph for sensor.hpp:



This graph shows which files directly or indirectly include this file:



Namespaces

- `cl`

Macros

- `#define CL_SENSOR`
- `#define CL_SENSOR_X(enum, value) enum = value,`
- `#define CL_SENSOR_X(enm, v) ::cl::Sensor::enm,`

Enumerations

- enum `cl::Sensor : std::uint64_t { CL_SENSOR_X, CL_SENSOR }`

Functions

- `std::ostream & operator<< (std::ostream &os, Sensor sensor)`

Variables

- `constexpr std::array< Sensor, 4 > cl::sensors`

7.71.1 Macro Definition Documentation

7.71.1.1 CL_SENSOR

```
#define CL_SENSOR
```

Value:

```
CL_SENSOR_X(LeftArm, 769) \
CL_SENSOR_X(Belly, 770) \
CL_SENSOR_X(RightArm, 771) \
CL_SENSOR_X(Chest, 772)
```

Definition at line 9 of file sensor.hpp.

7.71.1.2 CL_SENSOR_X [1/2]

```
#define CL_SENSOR_X(
    enm,
    v ) ::cl::Sensor::enm,
```

Definition at line 16 of file sensor.hpp.

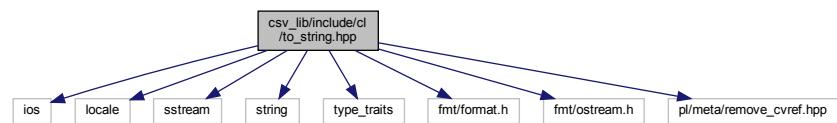
7.71.1.3 CL_SENSOR_X [2/2]

```
#define CL_SENSOR_X(
    enumerator,
    value ) enumerator = value,
```

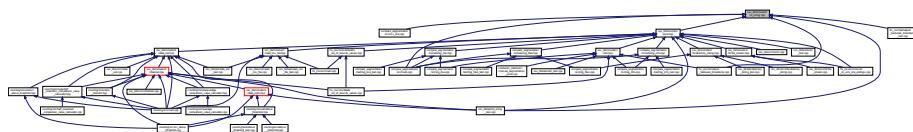
Definition at line 16 of file sensor.hpp.

7.72 csv_lib/include/cl/to_string.hpp File Reference

```
#include <iostream>
#include <locale>
#include <sstream>
#include <string>
#include <type_traits>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/meta/remove_cvref.hpp>
Include dependency graph for to_string.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

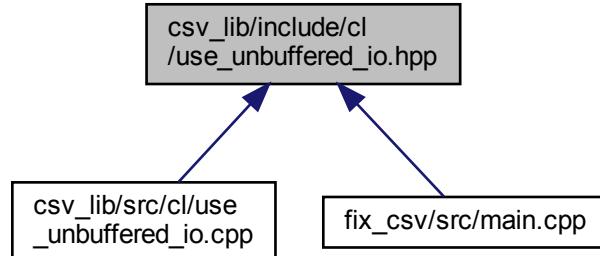
- `cl`

Functions

- template<typename Ty >
 `std::string cl::to_string (const Ty &ty)`

7.73 csv_lib/include/cl/use_unbuffered_io.hpp File Reference

This graph shows which files directly or indirectly include this file:



Namespaces

- `cl`

Functions

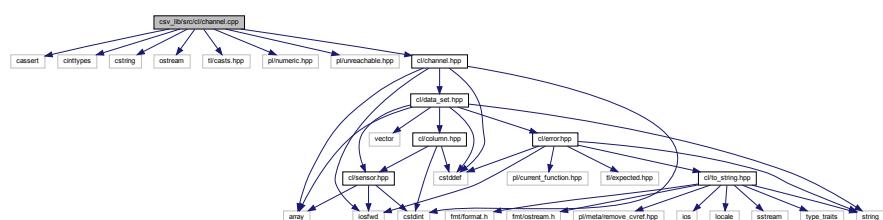
- void `cl::useUnbufferedIo ()`

7.74 csv_lib/src/cl/channel.cpp File Reference

```

#include <cassert>
#include <cinttypes>
#include <cstring>
#include <iostream>
#include <tl/casts.hpp>
#include <pl/numeric.hpp>
#include <pl/unreachable.hpp>
#include "cl/channel.hpp"
  
```

Include dependency graph for channel.cpp:



Namespaces

- `cl`

Macros

- `#define CL_CHANNEL_X(enm, v, acc) case Channel::enm: return data_set_accessor_v<Channel::enm>;`
- `#define CL_CHANNEL_X(enumerator, value, dataSetAccessor) case Channel::enumerator: return os << #enumerator;`

Functions

- `DataSet::ChannelAccessor cl::dataSetAccessor (Channel channel)`
- `std::ostream & cl::operator<< (std::ostream &os, Channel channel)`
- `bool cl::isAccelerometer (Channel channel)`
- `bool cl::isGyroscope (Channel channel)`
- `long double cl::threshold (Channel channel)`

7.74.1 Macro Definition Documentation

7.74.1.1 CL_CHANNEL_X [1/2]

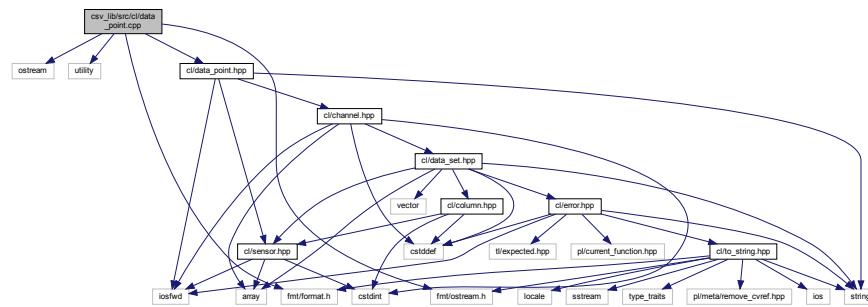
```
#define CL_CHANNEL_X(
    enm,
    v,
    acc ) case Channel::enm: return data_set_accessor_v<Channel::enm>;
```

7.74.1.2 CL_CHANNEL_X [2/2]

```
#define CL_CHANNEL_X(
    enumerator,
    value,
    dataSetAccessor ) case Channel::enumerator: return os << #enumerator;
```

7.75 csv_lib/src/cl/data_point.cpp File Reference

```
#include <iostream>
#include <utility>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include "cl/data_point.hpp"
Include dependency graph for data_point.cpp:
```



Namespaces

- `cl`

Functions

- `std::ostream & cl::operator<< (std::ostream &os, const DataPoint &dataPoint)`
- `dataPoint fileName ()`
- `dataPoint dataPoint time ()`
- `dataPoint dataPoint dataPoint sensor ()`
- `dataPoint dataPoint dataPoint dataPoint channel ()`
- `dataPoint dataPoint dataPoint dataPoint value ()`

7.75.1 Function Documentation

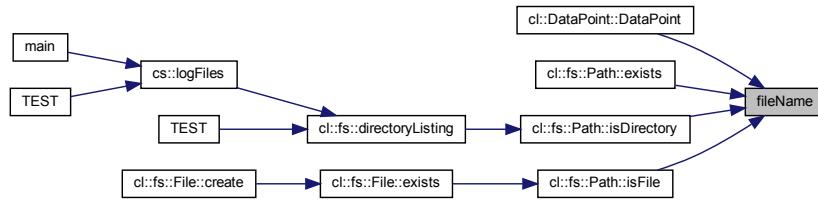
7.75.1.1 channel()

```
dataPoint dataPoint dataPoint dataPoint channel ( )
```

7.75.1.2 fileName()

dataPoint fileName ()

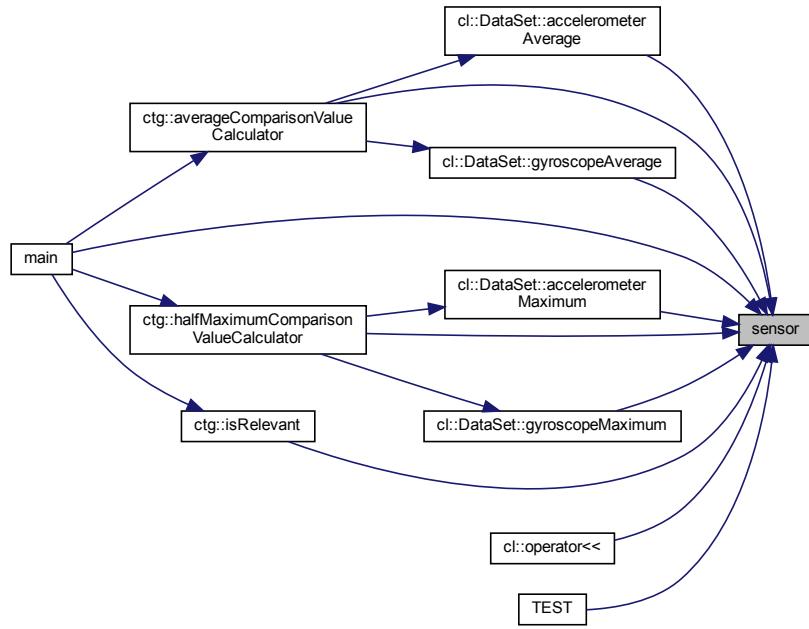
Here is the caller graph for this function:



7.75.1.3 sensor()

```
dataPoint dataPoint dataPoint sensor ( )
```

Here is the caller graph for this function:



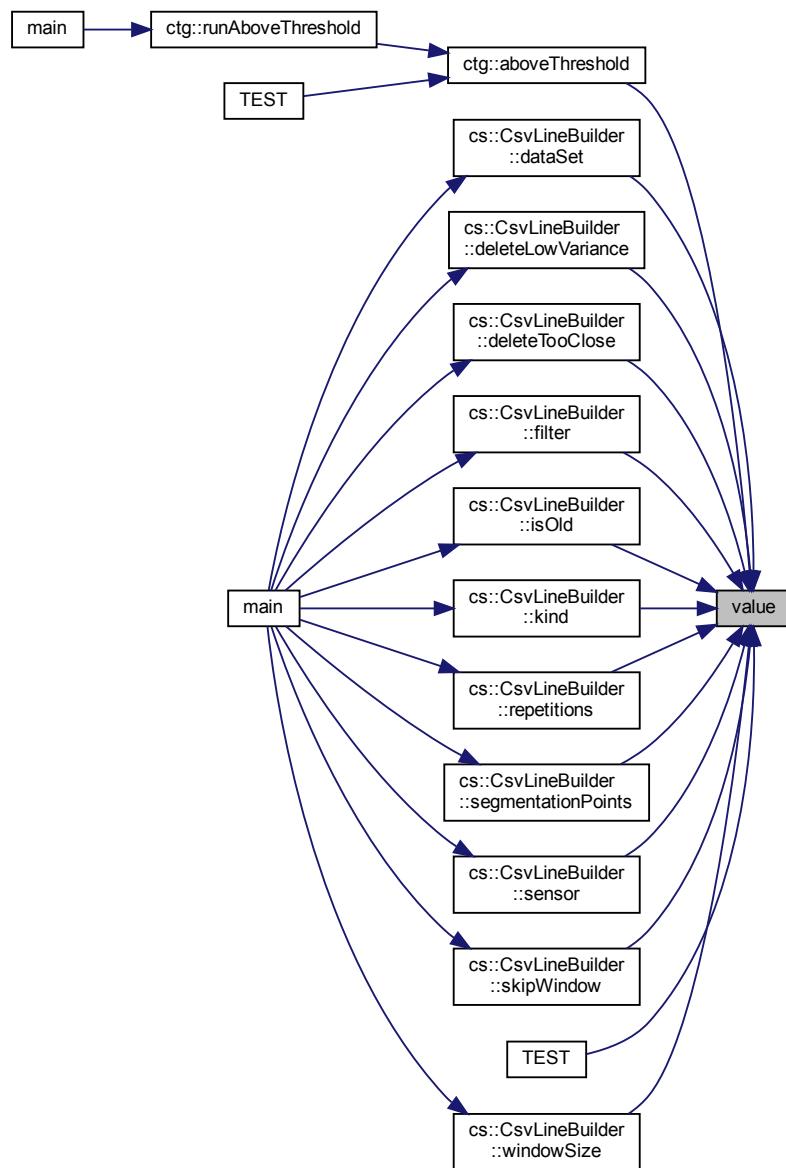
7.75.1.4 time()

```
dataPoint dataPoint time ( )
```

7.75.1.5 value()

```
dataPoint dataPoint dataPoint dataPoint value ( )
```

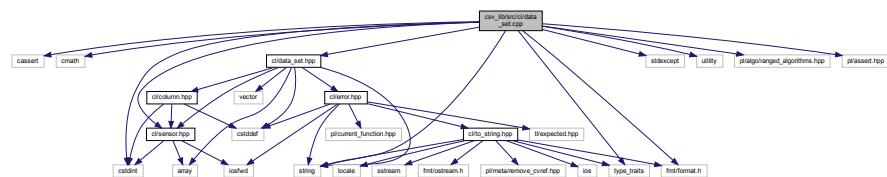
Here is the caller graph for this function:



7.76 csv_lib/src/cl/data_set.cpp File Reference

```
#include <cassert>
#include <cmath>
#include <cstdint>
#include <stdexcept>
#include <string>
#include <type_traits>
#include <utility>
#include <fmt/format.h>
#include <pl/algo/ranged_algorithms.hpp>
#include <pl/assert.hpp>
#include "cl/data_set.hpp"
#include "cl/sensor.hpp"

Include dependency graph for data_set.cpp:
```



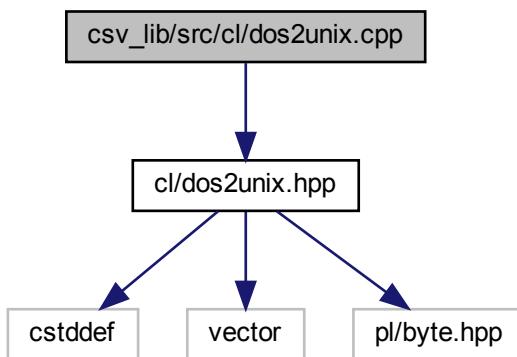
Namespaces

- `cl`

7.77 csv_lib/src/cl/dos2unix.cpp File Reference

```
#include "cl/dos2unix.hpp"

Include dependency graph for dos2unix.cpp:
```



Namespaces

- [cl](#)

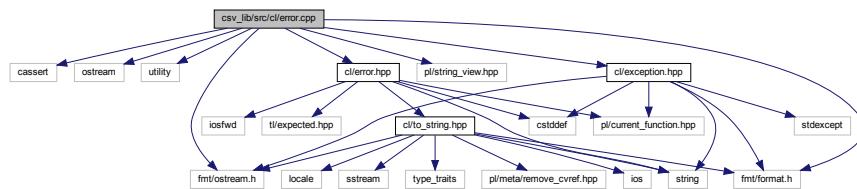
Functions

- `std::vector< pl::byte > cl::dos2unix (const void *p, std::size_t size)`

Converts DOS / Microsoft Windows line endings to UNIX line endings.

7.78 csv_lib/src/cl/error.cpp File Reference

```
#include <cassert>
#include <iostream>
#include <utility>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/string_view.hpp>
#include "cl/error.hpp"
#include "cl/exception.hpp"
Include dependency graph for error.cpp:
```



Namespaces

- [cl](#)

Macros

- `#define CL_ERROR_KIND_X(kind) case Error::kind: return #kind;`

Functions

- `std::ostream & cl::operator<< (std::ostream &os, const Error &error)`

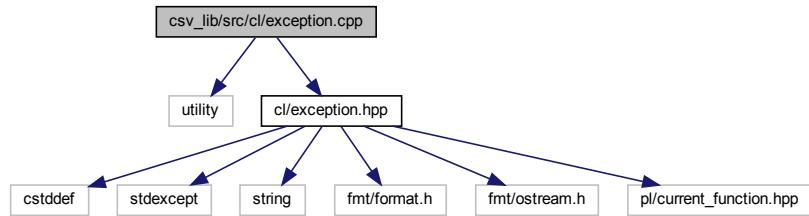
7.78.1 Macro Definition Documentation

7.78.1.1 CL_ERROR_KIND_X

```
#define CL_ERROR_KIND_X(
    kind ) case Error::kind:  return #kind;
```

7.79 csv_lib/src/cl/exception.cpp File Reference

```
#include <utility>
#include "cl/exception.hpp"
Include dependency graph for exception.cpp:
```

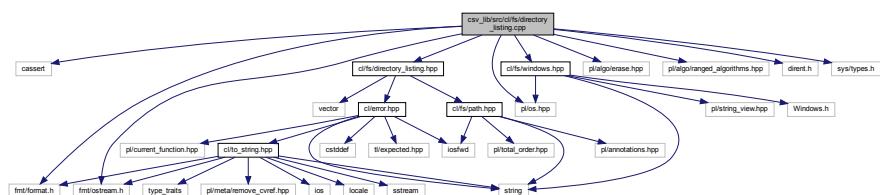


Namespaces

- `cl`

7.80 csv_lib/src/cl/fs/directory_listing.cpp File Reference

```
#include <cassert>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/algo/erase.hpp>
#include <pl/algo/ranged_algorithms.hpp>
#include <pl/os.hpp>
#include <cl/fs/windows.hpp>
#include <dirent.h>
#include <sys/types.h>
#include <cl/fs/directory_listing.hpp>
Include dependency graph for directory_listing.cpp:
```



Namespaces

- `cl`
- `cl::fs`

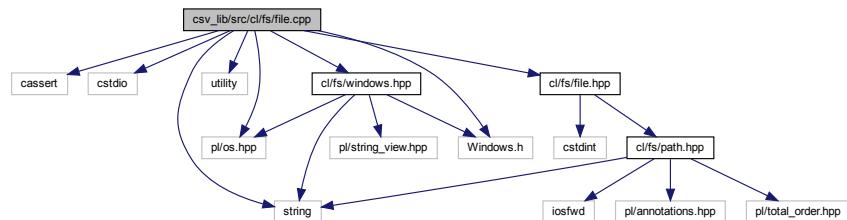
Functions

- Expected< std::vector< Path > > `cl::fs::directoryListing` (const Path &directoryPath, DirectoryListingOption directoryListingOption=DirectoryListingOption::ExcludeDotAndDotDot)

Creates a listing of the contents of a directory.

7.81 csv_lib/src/cl/fs/file.cpp File Reference

```
#include <cassert>
#include <cstdio>
#include <string>
#include <utility>
#include <pl/os.hpp>
#include "cl/fs/windows.hpp"
#include <Windows.h>
#include "cl/fs/file.hpp"
Include dependency graph for file.cpp:
```



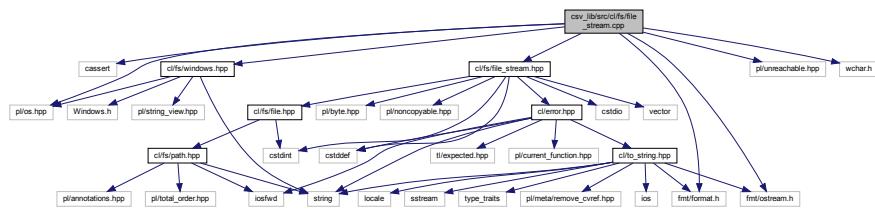
Namespaces

- `cl`
- `cl::fs`

7.82 csv_lib/src/cl/fs/file_stream.cpp File Reference

```
#include <cassert>
#include <pl/os.hpp>
#include <pl/unreachable.hpp>
#include "cl/fs/windows.hpp"
#include <wchar.h>
#include <fmt/format.h>
#include <fmt/ostream.h>
```

```
#include "cl/fs/file_stream.hpp"
Include dependency graph for file_stream.cpp:
```

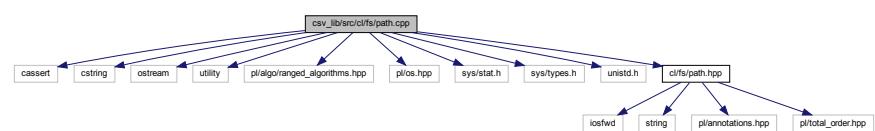


Namespaces

- `cl`
- `cl::fs`

7.83 csv_lib/src/cl/fs/path.cpp File Reference

```
#include <cassert>
#include <cstring>
#include <ostream>
#include <utility>
#include <p1/algo/ranged_algorithms.hpp>
#include <p1/os.hpp>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>
#include "cl/fs/path.hpp"
Include dependency graph for path.cpp:
```



Namespaces

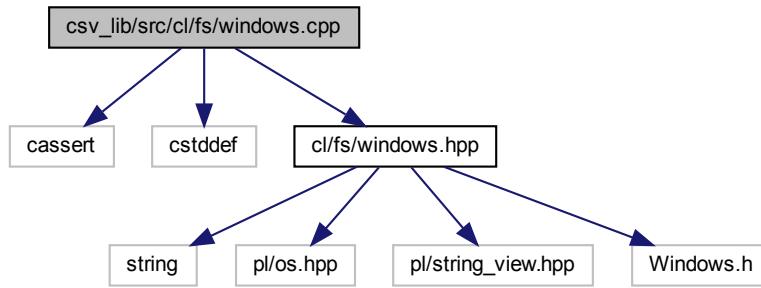
- `cl`
- `cl::fs`

Functions

- `std::ostream & cl::fs::operator<< (std::ostream &os, const Path &path)`
- `bool cl::fs::operator< (const Path &lhs, const Path &rhs) noexcept`
- `bool cl::fs::operator== (const Path &lhs, const Path &rhs) noexcept`

7.84 csv_lib/src/cl/fs/windows.cpp File Reference

```
#include <cassert>
#include <cstddef>
#include "cl/fs/windows.hpp"
Include dependency graph for windows.cpp:
```



Namespaces

- `cl`
- `cl::fs`

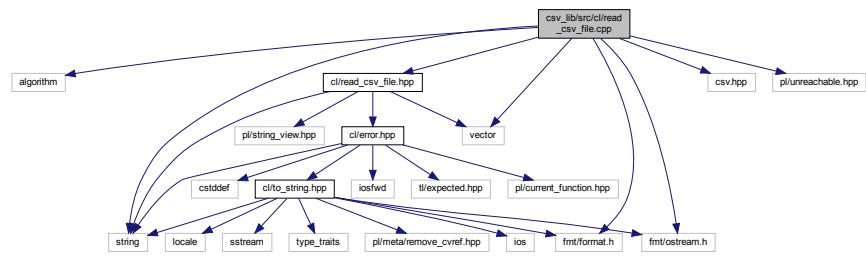
Functions

- `std::wstring cl::fs::utf8ToUtf16 (pl::string_view utf8)`
Converts a UTF-8 encoded string to a UTF-16 encoded wstring.
- `std::string cl::fs::utf16ToUtf8 (pl::wstring_view utf16)`
Converts a UTF-16 encoded wide character string to UTF-8 string.
- `std::wstring cl::fs::formatError (DWORD errorCode)`
Formats a WINAPI error code to a UTF-16 encoded wide character string.

7.85 csv_lib/src/cl/read_csv_file.cpp File Reference

```
#include <algorithm>
#include <string>
#include <vector>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <csv.hpp>
#include <pl/unreachable.hpp>
```

```
#include "cl/read_csv_file.hpp"
Include dependency graph for read_csv_file.cpp:
```



Namespaces

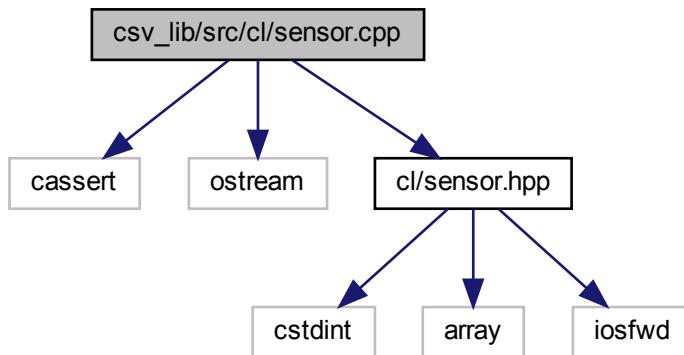
- `cl`

Functions

- `Expected< std::vector< std::vector< std::string > > > cl::readCsvFile(pl::string_view csvFilePath, std::vector< std::string > *columnNames=nullptr, CsvFileKind csvFileKind=CsvFileKind::Fixed) noexcept`

7.86 csv_lib/src/cl/sensor.cpp File Reference

```
#include <cassert>
#include <ostream>
#include "cl/sensor.hpp"
Include dependency graph for sensor.cpp:
```



Namespaces

- `cl`

Macros

- `#define CL_SENSOR_X(enumerator, value) case Sensor::enumerator: return os << #enumerator;`

Functions

- `std::ostream & cl::operator<< (std::ostream &os, Sensor sensor)`

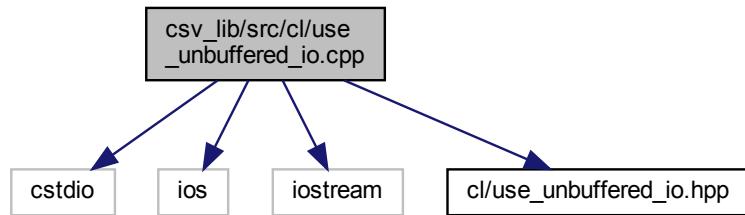
7.86.1 Macro Definition Documentation

7.86.1.1 CL_SENSOR_X

```
#define CL_SENSOR_X(
    enumerator,
    value ) case Sensor::enumerator: return os << #enumerator;
```

7.87 csv_lib/src/cl/use_unbuffered_io.cpp File Reference

```
#include <cstdio>
#include <iostream>
#include <iostream>
#include "cl/use_unbuffered_io.hpp"
Include dependency graph for use_unbuffered_io.cpp:
```



Namespaces

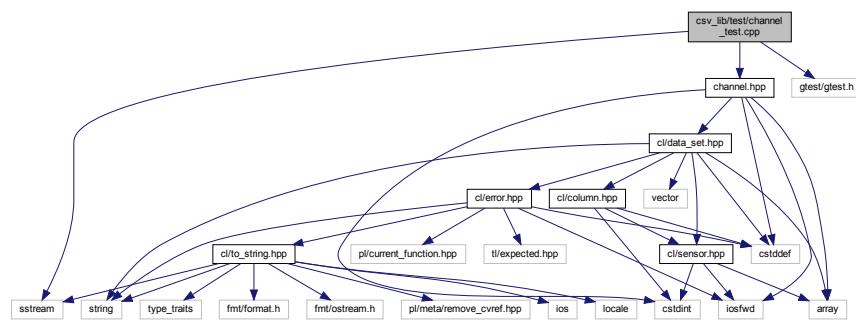
- `cl`

Functions

- `void cl::useUnbufferedIo ()`

7.88 csv_lib/test/channel_test.cpp File Reference

```
#include <sstream>
#include "gtest/gtest.h"
#include "channel.hpp"
Include dependency graph for channel_test.cpp:
```



Functions

- **TEST (channel, shouldHaveCorrectCount)**
 - **TEST (channel, shouldHaveCorrectValues)**
 - **TEST (channel, shouldPrintCorrectly)**
 - **TEST (channel, shouldMapToCorrectDataSetAccessors)**

7.88.1 Function Documentation

7.88.1.1 TEST() [1/4]

```
TEST (channel, shouldHaveCorrectCount)
```

Definition at line 7 of file channel.h.

7.88.1.2 TEST() [2/4]

```
TEST (channel, shouldHaveCorrectValues)
```

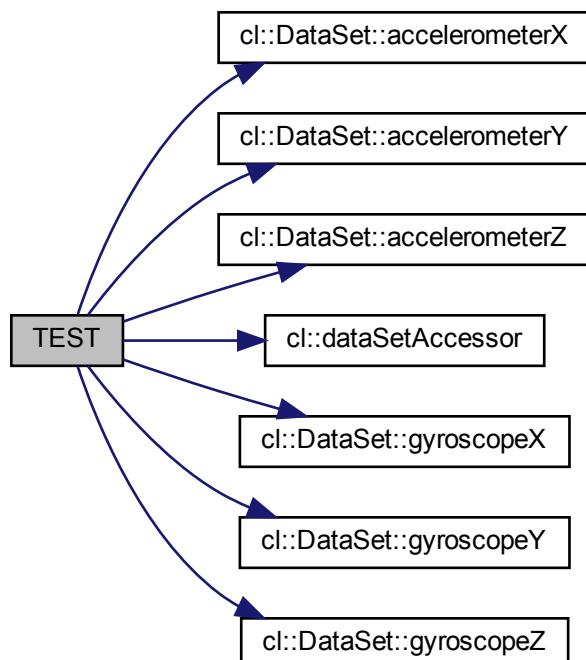
Definition at line 9 of file channel_test.cpp.

7.88.1.3 TEST() [3/4]

```
TEST (
    channel ,
    shouldMapToCorrectDataSetAccessors )
```

Definition at line 35 of file channel_test.cpp.

Here is the call graph for this function:



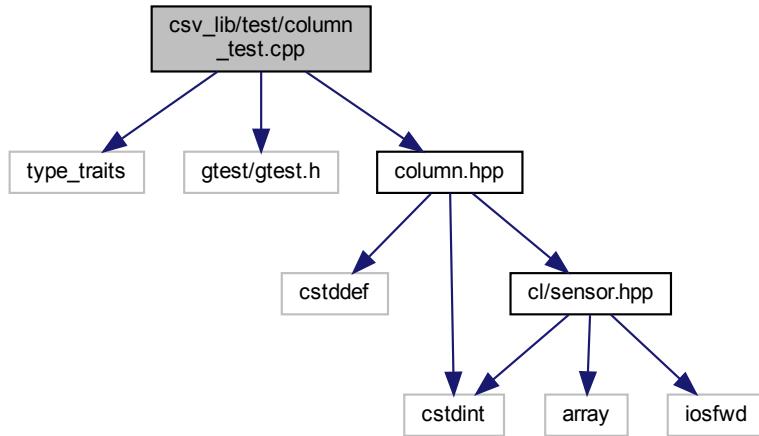
7.88.1.4 TEST() [4/4]

```
TEST (
    channel ,
    shouldPrintCorrectly )
```

Definition at line 19 of file channel_test.cpp.

7.89 csv_lib/test/column_test.cpp File Reference

```
#include <type_traits>
#include "gtest/gtest.h"
#include "column.hpp"
Include dependency graph for column_test.cpp:
```



Functions

- `TEST` (`column`, `shouldHaveCorrectIndex`)
- `TEST` (`column`, `shouldHaveCorrectColumnType`)

7.89.1 Function Documentation

7.89.1.1 TEST() [1/2]

```
TEST (
    column ,
    shouldHaveCorrectColumnType  )
```

Definition at line 22 of file `column_test.cpp`.

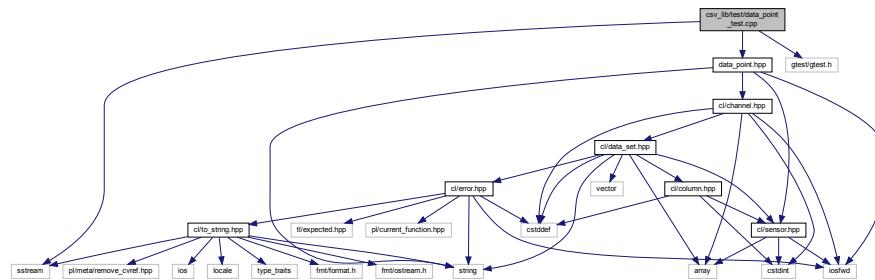
7.89.1.2 TEST() [2/2]

```
TEST (
    column ,
    shouldHaveCorrectIndex  )
```

Definition at line 7 of file `column_test.cpp`.

7.90 csv_lib/test/data_point_test.cpp File Reference

```
#include <sstream>
#include "gtest/gtest.h"
#include "data_point.hpp"
Include dependency graph for data_point_test.cpp:
```



Functions

- [TEST](#) (`DataPoint`, `shouldPrintCorrectly`)
- [TEST](#) (`DataPoint`, `shouldGetValuesCorrectly`)

Variables

- const `cl::DataPoint dp`

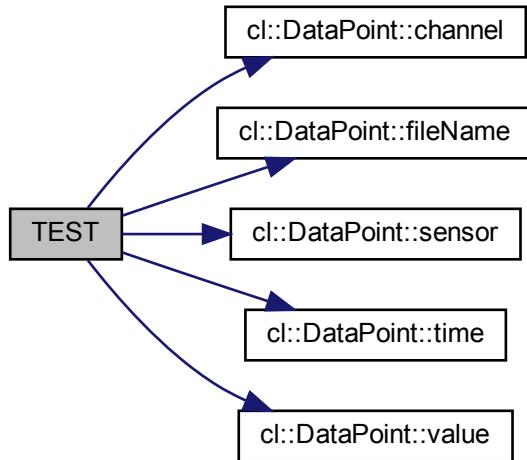
7.90.1 Function Documentation

7.90.1.1 TEST() [1/2]

```
TEST (
    DataPoint ,
    shouldGetValuesCorrectly )
```

Definition at line 23 of file `data_point_test.cpp`.

Here is the call graph for this function:



7.90.1.2 TEST() [2/2]

```
TEST (
    DataPoint ,
    shouldPrintCorrectly )
```

Definition at line 14 of file data_point_test.cpp.

7.90.2 Variable Documentation

7.90.2.1 dp

```
const cl::DataPoint dp
```

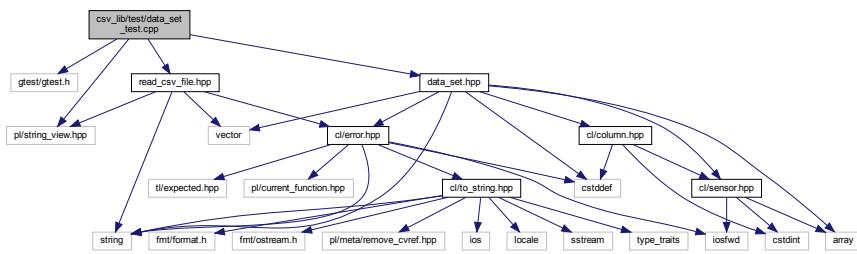
Initial value:

```
{
    "file.csv",
    0.01,
    cl::Sensor::Chest,
    cl::Channel::AccelerometerX,
    50.01}
```

Definition at line 7 of file data_point_test.cpp.

7.91 csv_lib/test/data_set_test.cpp File Reference

```
#include "gtest/gtest.h"
#include <pl/string_view.hpp>
#include "data_set.hpp"
#include "read_csv_file.hpp"
Include dependency graph for data_set_test.cpp:
```



Macros

- `#define EXPECT_LONG_DOUBLE_EQ(a, b) EXPECT_DOUBLE_EQ(static_cast<double>(a), static_cast<double>(b))`

Functions

- `TEST(DataSet, shouldBeAbleToCreateFromValidData)`
- `TEST(DataSet, shouldNotBeAbleToCreateFromEmptyMatrix)`
- `TEST(DataSet, shouldNotBeAbleToCreateFromJaggedMatrix)`
- `TEST(DataSet, shouldNotBeAbleToCreateFromInvalidData)`

7.91.1 Macro Definition Documentation

7.91.1.1 EXPECT_LONG_DOUBLE_EQ

```
#define EXPECT_LONG_DOUBLE_EQ(
    a,
    b ) EXPECT_DOUBLE_EQ(static_cast<double>(a), static_cast<double>(b))
```

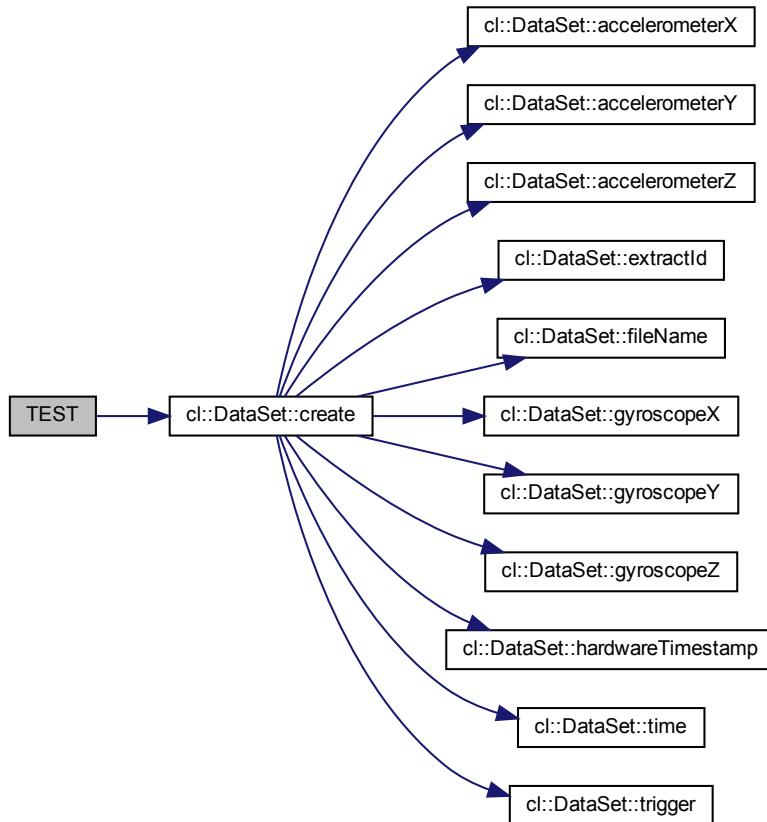
7.91.2 Function Documentation

7.91.2.1 TEST() [1/4]

```
TEST (
    DataSet ,
    shouldBeAbleToCreateFromValidData )
```

Definition at line 17 of file data_set_test.cpp.

Here is the call graph for this function:

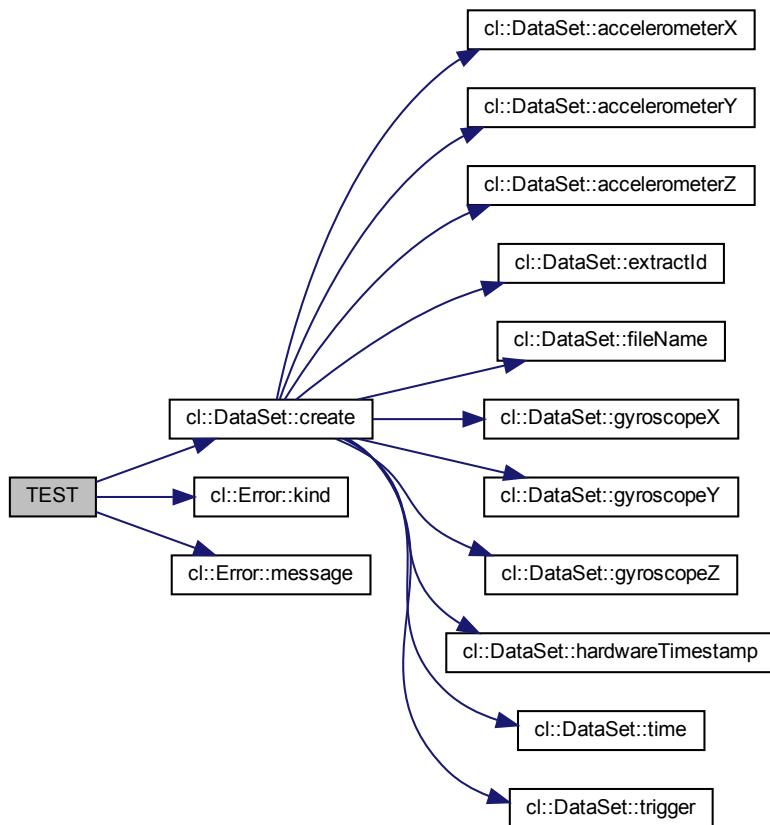


7.91.2.2 TEST() [2/4]

```
TEST (
    DataSet ,
    shouldNotBeAbleToCreateFromEmptyMatrix )
```

Definition at line 68 of file data_set_test.cpp.

Here is the call graph for this function:

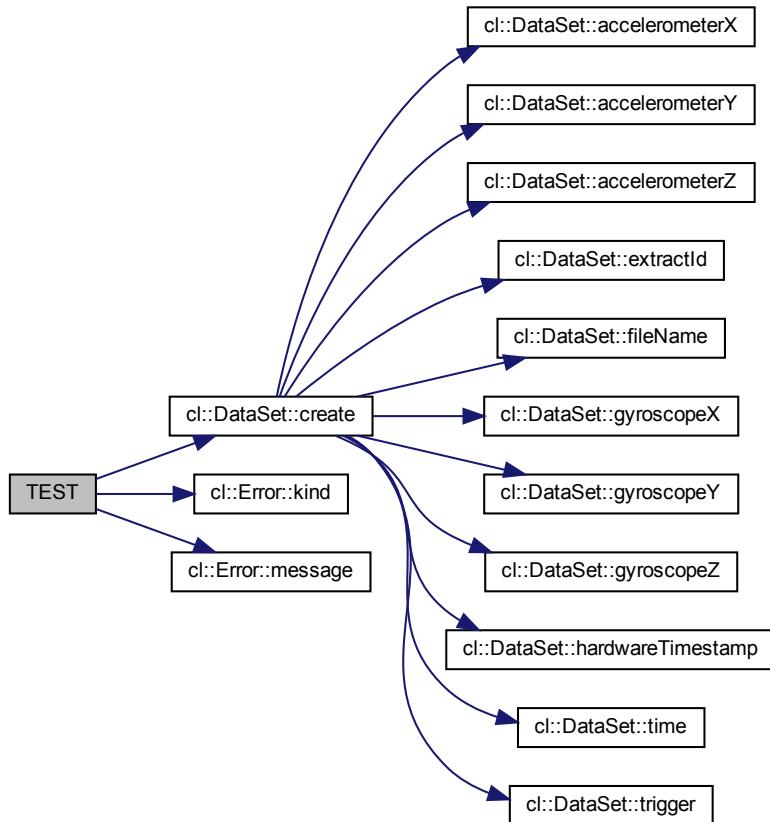


7.91.2.3 TEST() [3/4]

```
TEST (
    DataSet ,
    shouldNotBeAbleToCreateFromInvalidData )
```

Definition at line 108 of file `data_set_test.cpp`.

Here is the call graph for this function:

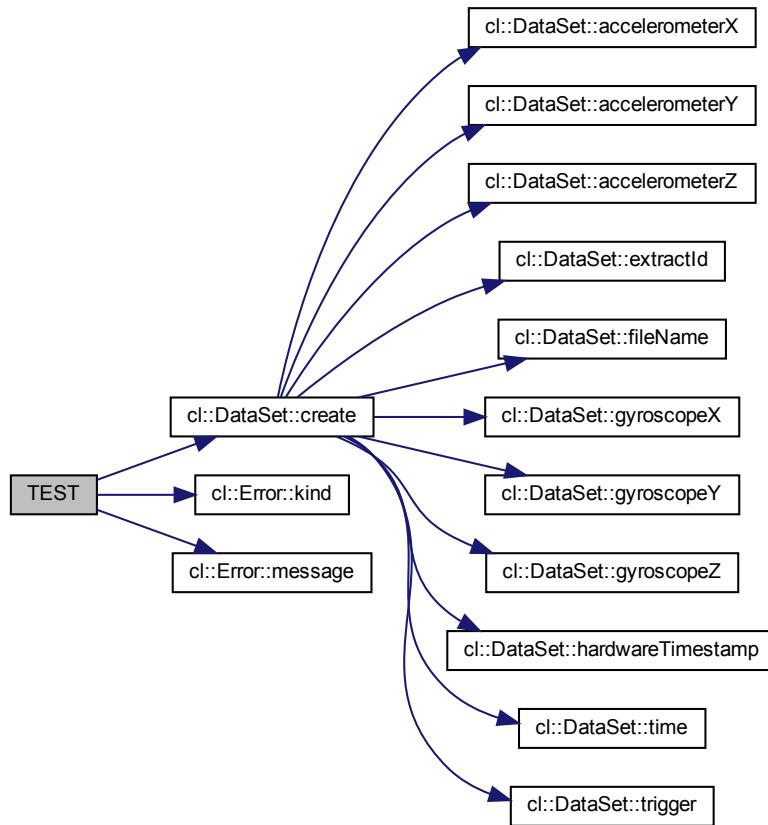


7.91.2.4 TEST() [4/4]

```
TEST (
    DataSet ,
    shouldNotBeAbleToCreateFromJaggedMatrix )
```

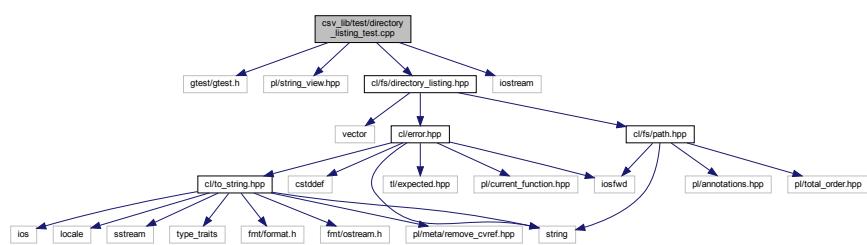
Definition at line 80 of file `data_set_test.cpp`.

Here is the call graph for this function:



7.92 csv_lib/test/directory_listing_test.cpp File Reference

```
#include "gtest/gtest.h"
#include <pl/string_view.hpp>
#include <cl/fs/directory_listing.hpp>
#include <iostream>
Include dependency graph for directory_listing_test.cpp:
```



Functions

- `TEST` (`directoryListing, shouldFindFiles`)
- `TEST` (`directoryListing, shouldFindFilesWithDotAndDotDot`)
- `TEST` (`directoryListing, shouldReturnErrorWhenPathDoesNotExist`)

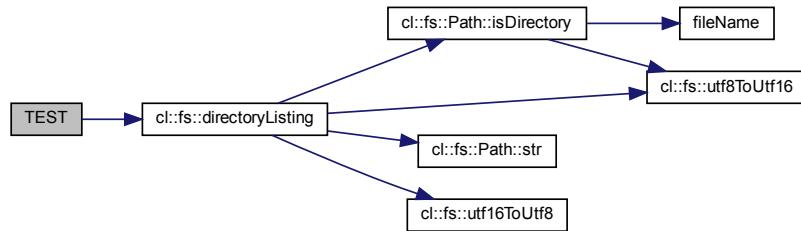
7.92.1 Function Documentation

7.92.1.1 TEST() [1/3]

```
TEST (
    directoryListing ,
    shouldFindFiles )
```

Definition at line 13 of file `directory_listing_test.cpp`.

Here is the call graph for this function:

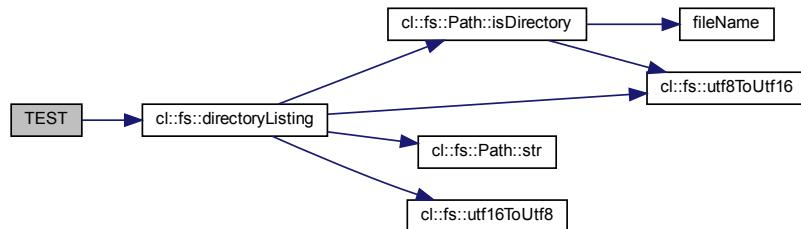


7.92.1.2 TEST() [2/3]

```
TEST (
    directoryListing ,
    shouldFindFilesWithDotAndDotDot )
```

Definition at line 28 of file `directory_listing_test.cpp`.

Here is the call graph for this function:

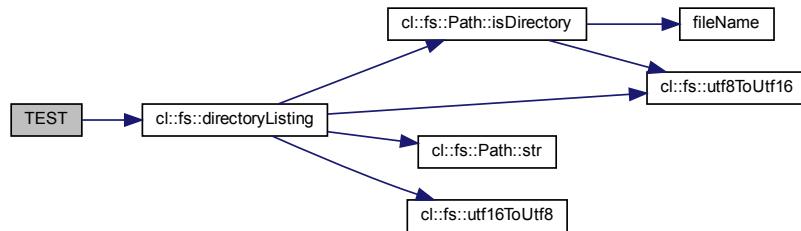


7.92.1.3 TEST() [3/3]

```
TEST (
    directoryListing ,
    shouldReturnErrorWhenPathDoesNotExist )
```

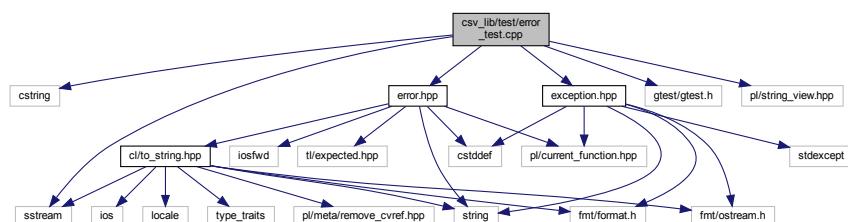
Definition at line 46 of file directory_listing_test.cpp.

Here is the call graph for this function:



7.93 csv_lib/test/error_test.cpp File Reference

```
#include <cstring>
#include <sstream>
#include "gtest/gtest.h"
#include <pl/string_view.hpp>
#include "error.hpp"
#include "exception.hpp"
Include dependency graph for error_test.cpp:
```



Functions

- [TEST \(error, shouldPrint\)](#)
- [TEST \(error, shouldReturnValues\)](#)
- [TEST \(error, shouldThrowExceptionWhenRaisesCalled\)](#)
- [TEST \(error, shouldCreateExpectedWithUnexpected\)](#)

Variables

- const `cl::Error error`

7.93.1 Function Documentation

7.93.1.1 TEST() [1/4]

```
TEST (
    error ,
    shouldCreateExpectedWithUnexpected )
```

Definition at line 59 of file error_test.cpp.

7.93.1.2 TEST() [2/4]

```
TEST (
    error ,
    shouldPrint )
```

Definition at line 19 of file error_test.cpp.

7.93.1.3 TEST() [3/4]

```
TEST (
    error ,
    shouldReturnValues )
```

Definition at line 29 of file error_test.cpp.

7.93.1.4 TEST() [4/4]

```
TEST (
    error ,
    shouldThrowExceptionWhenRaiseIsCalled )
```

Definition at line 37 of file error_test.cpp.

7.93.2 Variable Documentation

7.93.2.1 error

```
const cl::Error error
```

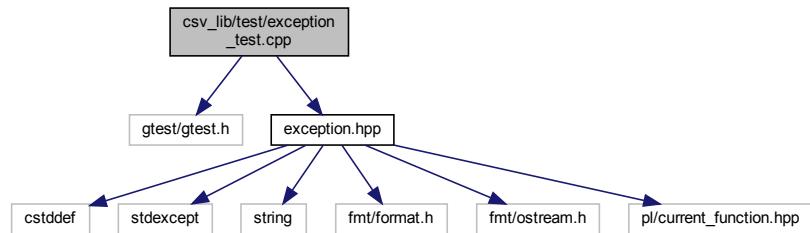
Initial value:

```
{
    cl::Error::Filesystem,
    "test_file.cpp",
    "bad_function",
    48,
    "Couldn't initialize the flux capacitor."}
```

Definition at line 12 of file error_test.cpp.

7.94 csv_lib/test/exception_test.cpp File Reference

```
#include "gtest/gtest.h"
#include "exception.hpp"
Include dependency graph for exception_test.cpp:
```



Functions

- [TEST](#) (exception, shouldWork)

7.94.1 Function Documentation

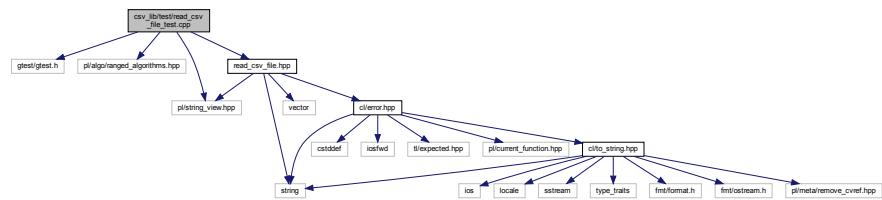
7.94.1.1 TEST()

```
TEST (
    exception ,
    shouldWork )
```

Definition at line 5 of file exception_test.cpp.

7.95 csv_lib/test/read_csv_file_test.cpp File Reference

```
#include "gtest/gtest.h"
#include <pl/algo/ranged_algorithms.hpp>
#include <pl/string_view.hpp>
#include "read_csv_file.hpp"
Include dependency graph for read_csv_file_test.cpp:
```



Functions

- [TEST](#) (`readCsvFile`, `shouldReadCsvFile`)
- [TEST](#) (`readCsvFile`, `shouldNotReadNonexistantCsvFile`)

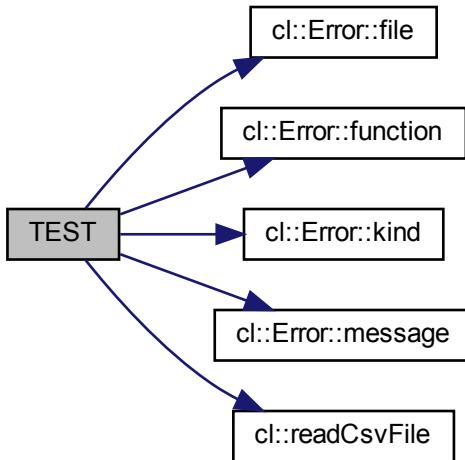
7.95.1 Function Documentation

7.95.1.1 TEST() [1/2]

```
TEST (
    readCsvFile ,
    shouldNotReadNonexistantCsvFile )
```

Definition at line 30 of file `read_csv_file_test.cpp`.

Here is the call graph for this function:



7.95.1.2 TEST() [2/2]

```
TEST (
    readCsvFile ,
    shouldReadCsvFile )
```

Definition at line 8 of file `read_csv_file_test.cpp`.

Here is the call graph for this function:

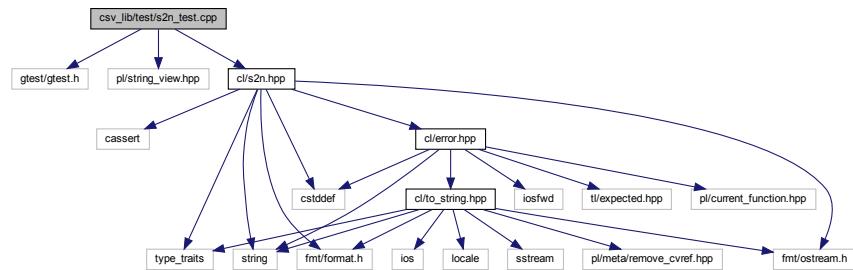


7.96 csv_lib/test/s2n_test.cpp File Reference

```
#include "gtest/gtest.h"
#include <pl/string_view.hpp>
```

```
#include "cl/s2n.hpp"
```

Include dependency graph for s2n_test.cpp:



Functions

- `TEST` (s2n, shouldWork)
- `TEST` (s2n, shouldReturnInvalidArgumentErrorIfInputIsInvalid)
- `TEST` (s2n, shouldReturnOutOfRangeErrorIfInputIsOutOfRange)

7.96.1 Function Documentation

7.96.1.1 TEST() [1/3]

```
TEST (
    s2n ,
    shouldReturnInvalidArgumentErrorIfInputIsInvalid )
```

Definition at line 21 of file s2n_test.cpp.

7.96.1.2 TEST() [2/3]

```
TEST (
    s2n ,
    shouldReturnOutOfRangeErrorIfInputIsOutOfRange )
```

Definition at line 29 of file s2n_test.cpp.

7.96.1.3 TEST() [3/3]

```
TEST (
    s2n ,
    shouldWork )
```

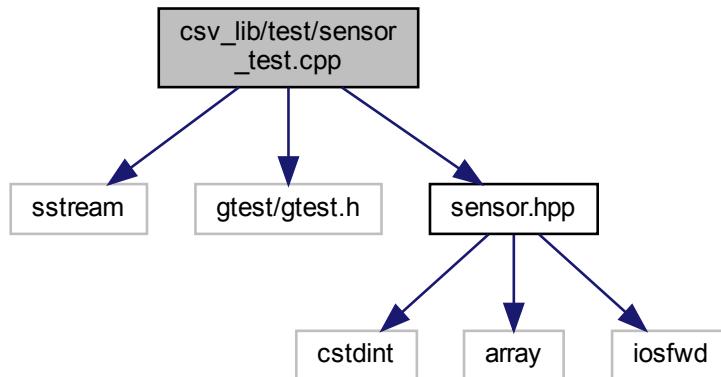
Definition at line 7 of file s2n_test.cpp.

Here is the call graph for this function:



7.97 csv_lib/test/sensor_test.cpp File Reference

```
#include <sstream>
#include "gtest/gtest.h"
#include "sensor.hpp"
Include dependency graph for sensor_test.cpp:
```



Functions

- [TEST \(sensor, shouldHaveCorrectValues\)](#)
- [TEST \(sensor, shouldPrintCorrely\)](#)

7.97.1 Function Documentation

7.97.1.1 TEST() [1/2]

```
TEST (
    sensor ,
    shouldHaveCorrectValues )
```

Definition at line 7 of file sensor_test.cpp.

7.97.1.2 TEST() [2/2]

```
TEST (
    sensor ,
    shouldPrintCorrectly )
```

Definition at line 15 of file sensor_test.cpp.

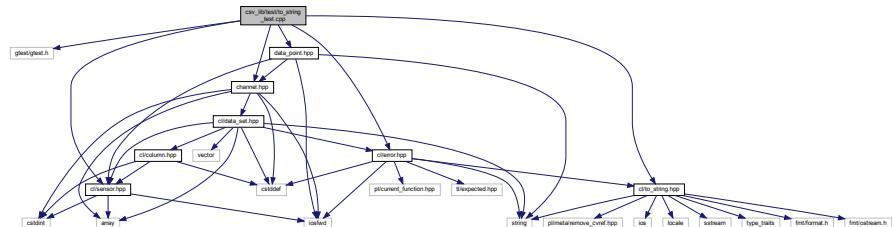
Here is the call graph for this function:



7.98 csv_lib/test/to_string_test.cpp File Reference

```
#include "gtest/gtest.h"
#include "channel.hpp"
#include "data_point.hpp"
#include "error.hpp"
#include "sensor.hpp"
#include "to_string.hpp"
```

Include dependency graph for to_string_test.cpp:



Functions

- [TEST](#) (to_string, test)

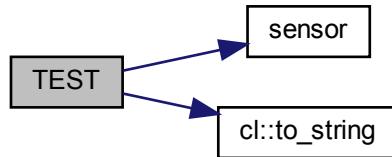
7.98.1 Function Documentation

7.98.1.1 TEST()

```
TEST (
    to_string ,
    test )
```

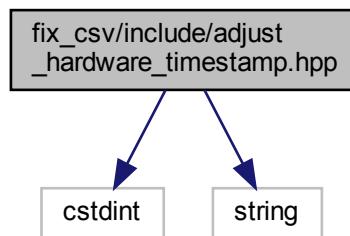
Definition at line 9 of file `to_string_test.cpp`.

Here is the call graph for this function:

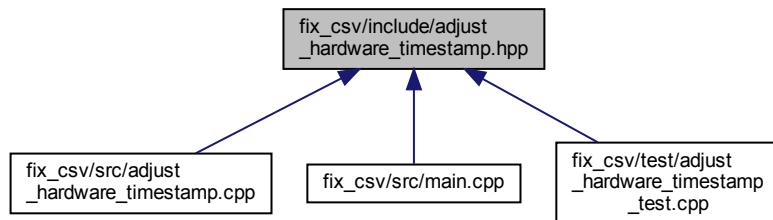


7.99 fix_csv/include/adjust_hw_timestamp.hpp File Reference

```
#include <cstdint>
#include <string>
Include dependency graph for adjust_hw_timestamp.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

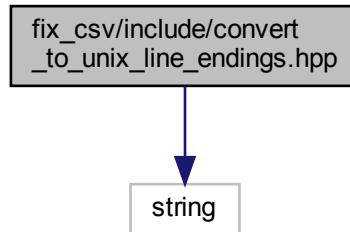
- `fmc`

Functions

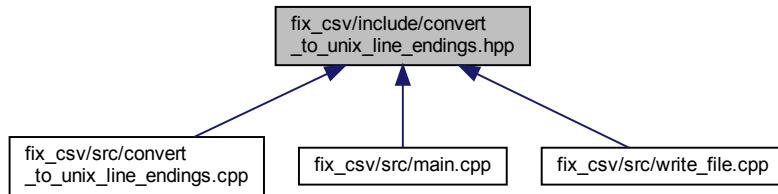
- void `fmc::adjustHardwareTimestamp` (`std::string *cellContent, const std::string &nextRowHardwareTimestamp, std::uint64_t *overflowCount)`

7.100 fix_csv/include/convert_to_unix_line_endings.hpp File Reference

```
#include <string>
Include dependency graph for convert_to_unix_line_endings.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

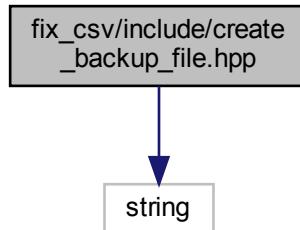
- [fmc](#)

Functions

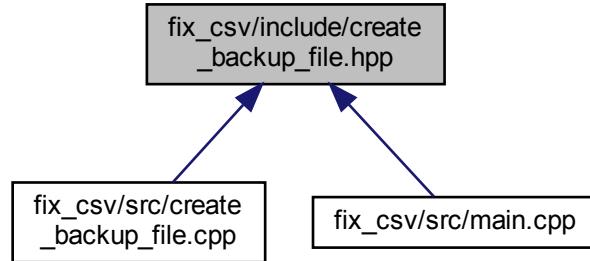
- bool [fmc::convertToUnixLineEndings](#) (const std::string &csvPath)

7.101 fix_csv/include/create_backup_file.hpp File Reference

```
#include <string>
Include dependency graph for create_backup_file.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

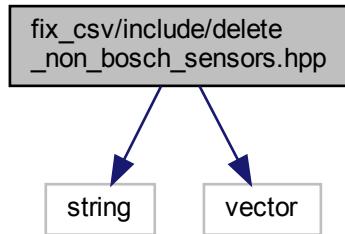
- [fmc](#)

Functions

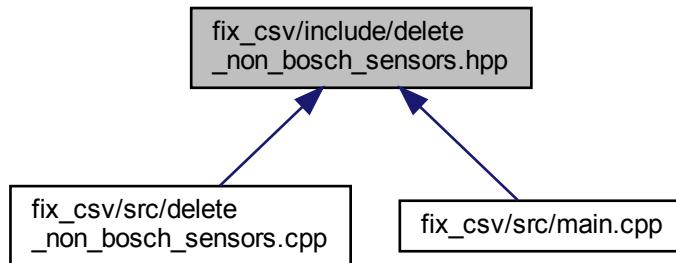
- `bool fmc::createBackupFile (const std::string &csvFilePath, const std::string &backupFilePath)`

7.102 fix_csv/include/delete_non_bosch_sensors.hpp File Reference

```
#include <string>
#include <vector>
Include dependency graph for delete_non_bosch_sensors.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

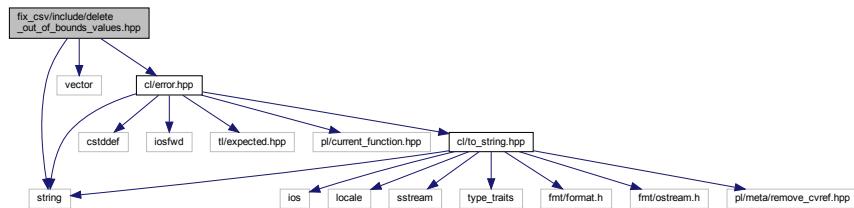
- `fmc`

Functions

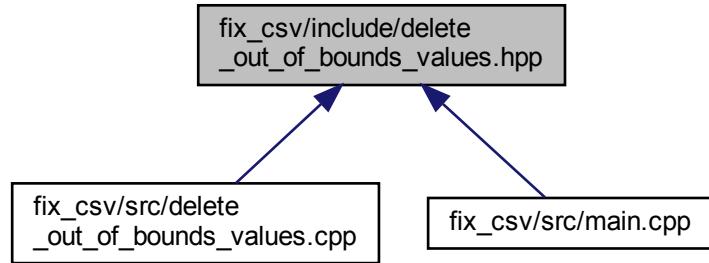
- void `fmc::deleteNonBoschSensors` (`std::vector< std::vector< std::string >> *data)`

7.103 fix_csv/include/delete_out_of_bounds_values.hpp File Reference

```
#include <string>
#include <vector>
#include "cl/error.hpp"
Include dependency graph for delete_out_of_bounds_values.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

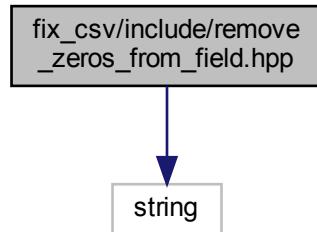
- fmc

Functions

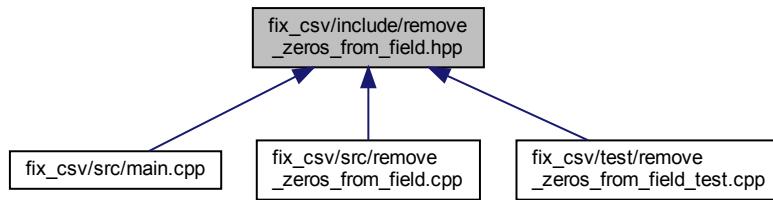
- `cl::Expected< void > fmc::deleteOutOfBoundsValues (std::vector< std::vector< std::string >> *data)`

7.104 fix_csv/include/remove_zeros_from_field.hpp File Reference

```
#include <string>
Include dependency graph for remove_zeros_from_field.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

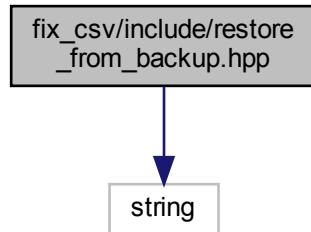
- [fmc](#)

Functions

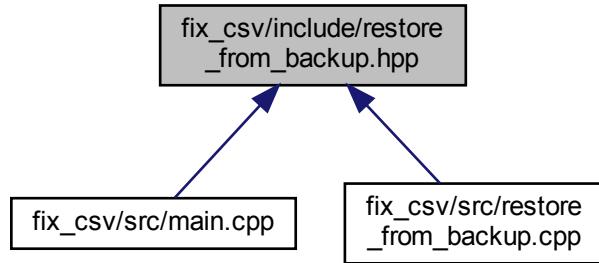
- void [fmc::removeZerosFromField](#) (std::string *field)

7.105 fix_csv/include/restore_from_backup.hpp File Reference

```
#include <string>
Include dependency graph for restore_from_backup.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

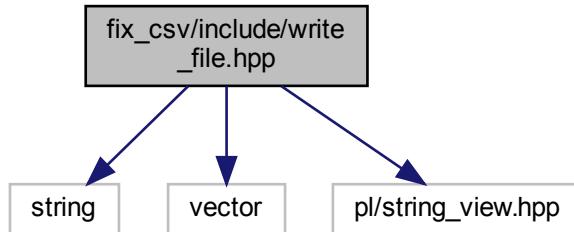
- `fmc`

Functions

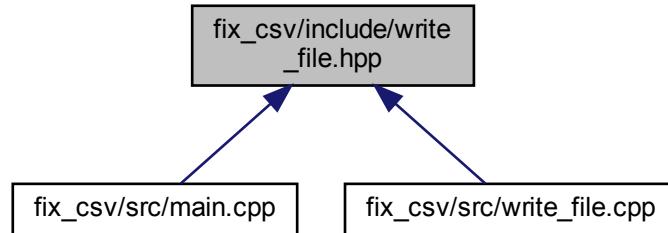
- bool `fmc::restoreFromBackup` (const std::string &csvFilePath, const std::string &backupFilePath)

7.106 fix_csv/include/write_file.hpp File Reference

```
#include <string>
#include <vector>
#include <pl/string_view.hpp>
Include dependency graph for write_file.hpp:
```



This graph shows which files directly or indirectly include this file:



Namespaces

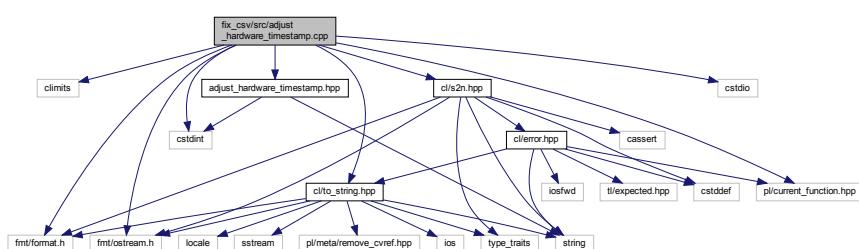
- [fmc](#)

Functions

- `bool fmc::writeFile (pl::string_view csvPath, pl::string_view csvFileExtension, const std::vector< std::string > &columnNames, const std::vector< std::vector< std::string >> &data)`

7.107 fix_csv/src/adjust_hardware_timestamp.cpp File Reference

```
#include <climits>
#include <cstdint>
#include <cstdio>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/current_function.hpp>
#include "cl/s2n.hpp"
#include "cl/to_string.hpp"
#include "adjust_hardware_timestamp.hpp"
Include dependency graph for adjust_hardware_timestamp.cpp:
```



Namespaces

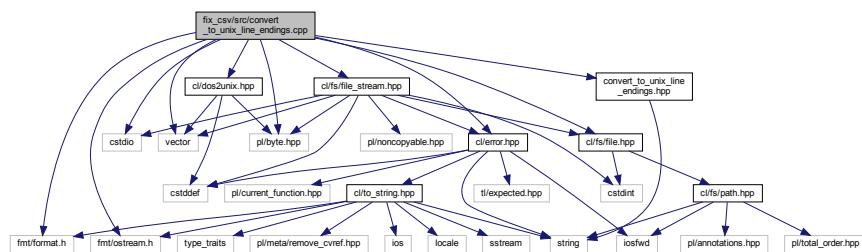
- fmc

Functions

- void [fmc::adjustHardwareTimestamp](#) (std::string *cellContent, const std::string &nextRowHardwareTimestamp, std::uint64_t *overflowCount)

7.108 fix_csv/src/convert_to_unix_line_endings.cpp File Reference

```
#include <cstdio>
#include <vector>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/byte.hpp>
#include "cl/dos2unix.hpp"
#include "cl/error.hpp"
#include "cl/fs/file.hpp"
#include "cl/fs/file_stream.hpp"
#include "convert_to_unix_line_endings.hpp"
Include dependency graph for convert_to_unix_line_endings.cpp:
```



Namespaces

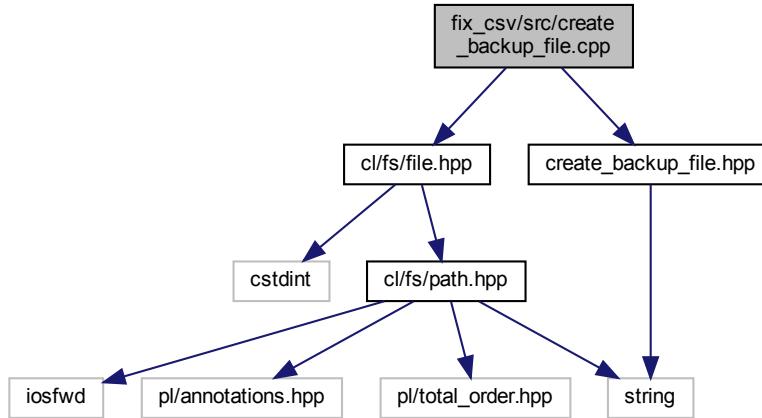
- fmc

Functions

- bool [fmc::convertToUnixLineEndings](#) (const std::string &csvPath)

7.109 fix_csv/src/create_backup_file.cpp File Reference

```
#include "cl/fs/file.hpp"
#include "create_backup_file.hpp"
Include dependency graph for create_backup_file.cpp:
```



Namespaces

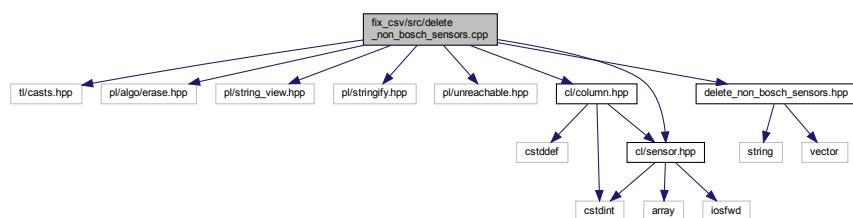
- `fmc`

Functions

- `bool fmc::createBackupFile (const std::string &csvFilePath, const std::string &backupFilePath)`

7.110 fix_csv/src/delete_non_bosch_sensors.cpp File Reference

```
#include <t1/casts.hpp>
#include <pl/algo/erase.hpp>
#include <pl/string_view.hpp>
#include <pl/stringify.hpp>
#include <pl/unreachable.hpp>
#include "cl/column.hpp"
#include "cl/sensor.hpp"
#include "delete_non_bosch_sensors.hpp"
Include dependency graph for delete_non_bosch_sensors.cpp:
```



Namespaces

- [fmc](#)

Macros

- `#define CL_SENSOR_X(enm, value) case cl::Sensor::enm: return PL_STRINGIFY(value);`

Functions

- `void fmc::deleteNonBoschSensors (std::vector< std::vector< std::string >> *data)`

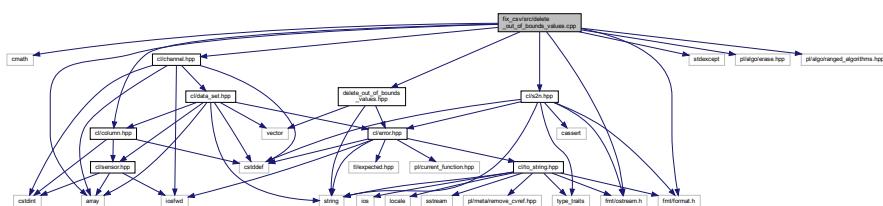
7.110.1 Macro Definition Documentation

7.110.1.1 CL_SENSOR_X

```
#define CL_SENSOR_X(
    enm,
    value ) case cl::Sensor::enm: return PL_STRINGIFY(value);
```

7.111 fix_csv/src/delete_out_of_bounds_values.cpp File Reference

```
#include <cmath>
#include <array>
#include <stdexcept>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/algo/erase.hpp>
#include <pl/algo/ranged_algorithms.hpp>
#include "cl/channel.hpp"
#include "cl/column.hpp"
#include "cl/s2n.hpp"
#include "delete_out_of_bounds_values.hpp"
Include dependency graph for delete_out_of_bounds_values.cpp:
```



Namespaces

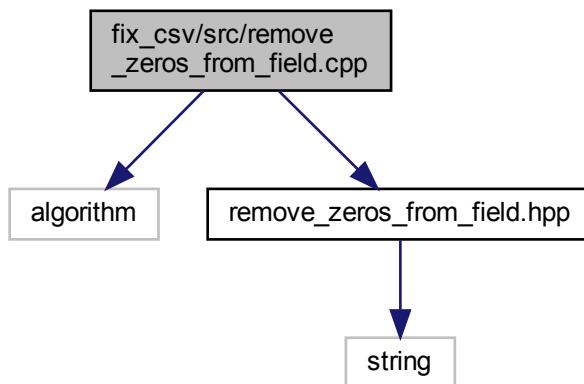
- [fmc](#)

Functions

- `cl::Expected< void > fmc::deleteOutOfBoundsValues (std::vector< std::vector< std::string >> *data)`

7.112 fix_csv/src/remove_zeros_from_field.cpp File Reference

```
#include <algorithm>
#include "remove_zeros_from_field.hpp"
Include dependency graph for remove_zeros_from_field.cpp:
```



Namespaces

- `fmc`

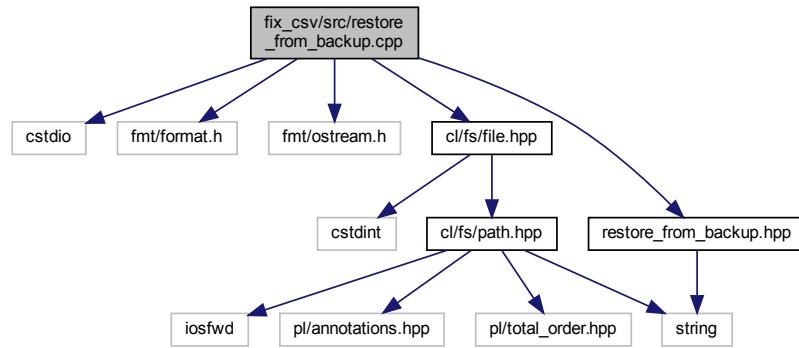
Functions

- `void fmc::removeZerosFromField (std::string *field)`

7.113 fix_csv/src/restore_from_backup.cpp File Reference

```
#include <cstdio>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include "cl/fs/file.hpp"
```

```
#include "restore_from_backup.hpp"
Include dependency graph for restore_from_backup.cpp:
```



Namespaces

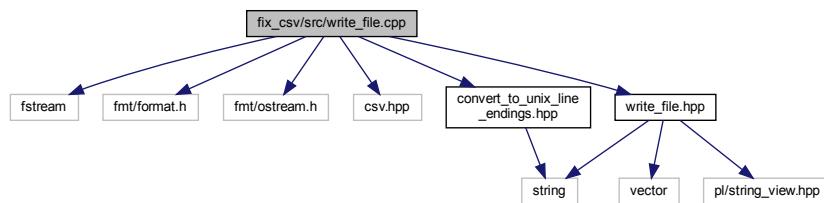
- `fmc`

Functions

- `bool fmc::restoreFromBackup (const std::string &csvFilePath, const std::string &backupFilePath)`

7.114 fix_csv/src/write_file.cpp File Reference

```
#include <fstream>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <csv.hpp>
#include "convert_to_unix_line_endings.hpp"
#include "write_file.hpp"
Include dependency graph for write_file.cpp:
```



Namespaces

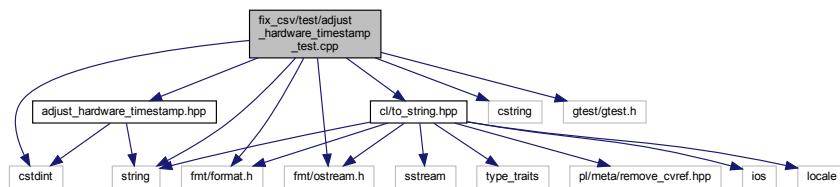
- `fmc`

Functions

- bool `fmc::writeFile` (pl::string_view csvPath, pl::string_view csvFileExtension, const std::vector< std::string > &columnNames, const std::vector< std::vector< std::string >> &data)

7.115 fix_csv/test/adjust_hwre_timestamp_test.cpp File Reference

```
#include <cstdint>
#include <cstring>
#include <string>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include "gtest/gtest.h"
#include "cl/to_string.hpp"
#include "adjust_hwre_timestamp.hpp"
Include dependency graph for adjust_hwre_timestamp_test.cpp:
```



Functions

- `TEST` (`adjustHardwareTimestamp`, `shouldDoNothingForNonOverflowedValue`)
- `TEST` (`adjustHardwareTimestamp`, `shouldIncrementOverflowCount`)
- `TEST` (`adjustHardwareTimestamp`, `shouldWorkForOneRoundOfOverflow`)
- `TEST` (`adjustHardwareTimestamp`, `shouldWorkForTwoRoundsOfOverflow`)
- `TEST` (`adjustHardwareTimestamp`, `shouldWork`)

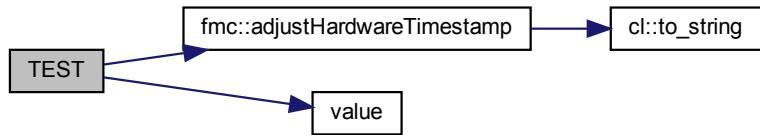
7.115.1 Function Documentation

7.115.1.1 TEST() [1/5]

```
TEST (
    adjustHardwareTimestamp ,
    shouldDoNothingForNonOverflowedValue )
```

Definition at line 15 of file `adjust_hwre_timestamp_test.cpp`.

Here is the call graph for this function:



7.115.1.2 TEST() [2/5]

```
TEST (
    adjustHardwareTimestamp ,
    shouldIncrementOverflowCount )
```

Definition at line 26 of file `adjust_hardware_timestamp_test.cpp`.

Here is the call graph for this function:

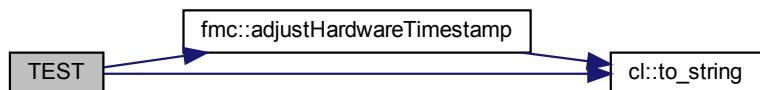


7.115.1.3 TEST() [3/5]

```
TEST (
    adjustHardwareTimestamp ,
    shouldWork )
```

Definition at line 132 of file `adjust_hardware_timestamp_test.cpp`.

Here is the call graph for this function:



7.115.1.4 TEST() [4/5]

```
TEST (
    adjustHardwareTimestamp ,
    shouldWorkForOneRoundOfOverflow )
```

Definition at line 48 of file `adjust_hardware_timestamp_test.cpp`.

Here is the call graph for this function:

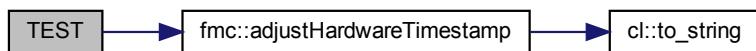


7.115.1.5 TEST() [5/5]

```
TEST (
    adjustHardwareTimestamp ,
    shouldWorkForTwoRoundsOfOverflow )
```

Definition at line 96 of file `adjust_hardware_timestamp_test.cpp`.

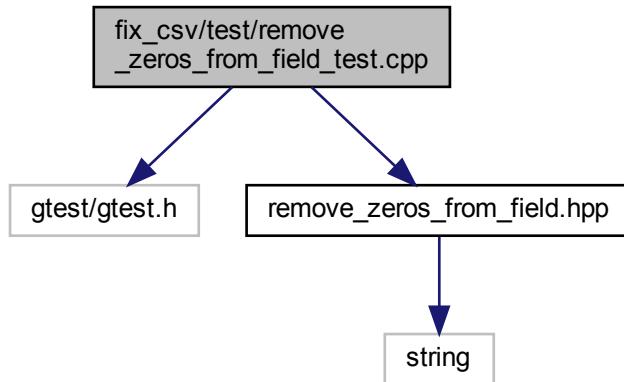
Here is the call graph for this function:



7.116 fix_csv/test/remove_zeros_from_field_test.cpp File Reference

```
#include "gtest/gtest.h"
#include "remove_zeros_from_field.hpp"
```

Include dependency graph for remove_zeros_from_field_test.cpp:



Functions

- [TEST \(removeZerosFromField, shouldRemoveDotAndZeros\)](#)
- [TEST \(removeZerosFromField, shouldNotRemovelfNonZerosFollow\)](#)
- [TEST \(removeZerosFromField, shouldNotRemovelfNoDot\)](#)
- [TEST \(removeZerosFromField, shouldDoNothingIfStringIsEmpty\)](#)
- [TEST \(removeZerosFromField, shouldDeleteStringIfStringIsSingleDot\)](#)
- [TEST \(removeZerosFromField, shouldDeleteStringIfStringIsDotAndZero\)](#)

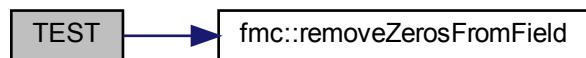
7.116.1 Function Documentation

7.116.1.1 TEST() [1/6]

```
TEST (
    removeZerosFromField ,
    shouldDeleteStringIfStringIsDotAndZero )
```

Definition at line 53 of file remove_zeros_from_field_test.cpp.

Here is the call graph for this function:

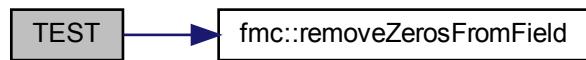


7.116.1.2 TEST() [2/6]

```
TEST (
    removeZerosFromField ,
    shouldDeleteStringIfStringIsSingleDot )
```

Definition at line 44 of file remove_zeros_from_field_test.cpp.

Here is the call graph for this function:



7.116.1.3 TEST() [3/6]

```
TEST (
    removeZerosFromField ,
    shouldDoNothingIfStringIsEmpty )
```

Definition at line 35 of file remove_zeros_from_field_test.cpp.

Here is the call graph for this function:



7.116.1.4 TEST() [4/6]

```
TEST (
    removeZerosFromField ,
    shouldNotRemoveIfNoDot )
```

Definition at line 25 of file remove_zeros_from_field_test.cpp.

Here is the call graph for this function:

**7.116.1.5 TEST() [5/6]**

```
TEST (
    removeZerosFromField ,
    shouldNotRemoveIfNonZerosFollow )
```

Definition at line 15 of file remove_zeros_from_field_test.cpp.

Here is the call graph for this function:



7.116.1.6 TEST() [6/6]

```
TEST (
    removeZerosFromField ,
    shouldRemoveDotAndZeros )
```

Definition at line 5 of file remove_zeros_from_field_test.cpp.

Here is the call graph for this function:



Index

~FileStream
 cl::fs::FileStream, 96

above_threshold.cpp
 channel, 174
 channelAccessor, 175
 CL_CHANNEL_X, 174

above_threshold_test.cpp
 EXPECT_LONG_DOUBLE_EQ, 177
 TEST, 178

aboveThreshold
 ctg, 43

accelerometerAverage
 cl::DataSet, 72

accelerometerMaximum
 cl::DataSet, 72

accelerometerThreshold
 cl, 22

AccelerometerX
 cl, 13

accelerometerX
 cl::DataSet, 73

AccelerometerY
 cl, 13

accelerometerY
 cl::DataSet, 73

AccelerometerZ
 cl, 13

accelerometerZ
 cl::DataSet, 74

adjust_hardware_timestamp_test.cpp
 TEST, 248–250

adjustHardwareTimestamp
 fmc, 48

Andre_1
 cm, 31

Andre_2
 cm, 31

Andre_3
 cm, 31

Andre_Squats_1
 cm, 31

Andre_Squats_2
 cm, 31

asMilliseconds
 cm::ManualSegmentationPoint, 110

averageComparisonValueCalculator
 ctg, 44

base_type

 cl::Exception, 86

Both
 cs, 33

build
 cs::CsvLineBuilder, 55

Butterworth
 cs, 33

Channel
 cl, 13

channel

- above_threshold.cpp, 174
- cl::DataPoint, 68
- data_point.cpp, 204

channel.cpp
 CL_CHANNEL_X, 203

channel.hpp
 CL_CHANNEL, 181
 CL_CHANNEL_X, 182

channel_test.cpp
 TEST, 215, 216

ChannelAccessor
 cl::DataSet, 72

channelAccessor
 above_threshold.cpp, 175

channelCount
 cl, 23

channels
 cl, 23

cl, 11

- accelerometerThreshold, 22
- AccelerometerX, 13
- AccelerometerY, 13
- AccelerometerZ, 13
- Channel, 13
- channelCount, 23
- channels, 23
- CL_CHANNEL, 13
- CL_CHANNEL_X, 13
- CL_SENSOR, 14
- CL_SENSOR_X, 14
- CL_SPECIALIZE_COL_TRAITS, 14–16
- Column, 13
- column_index, 23
- column_type, 12
- CsvFileKind, 13
- data_set_accessor_v, 23
- dataSetAccessor, 16
- dos2unix, 16
- Expected, 12

ExtractId, 13
 Fixed, 14
 gyroscopeThreshold, 23
 GyroscopeX, 13
 GyroscopeY, 13
 GyroscopeZ, 13
 HardwareTimestamp, 13
 isAccelerometer, 17
 isGyroscope, 17
 operator<<, 18, 19
 Raw, 14
 readCsvFile, 20
 s2n, 20
 SamplingRate, 13
 Sensor, 14
 sensors, 24
 threshold, 21
 Time, 13
 to_string, 21
 Trigger, 13
 useUnbufferedIo, 22
 cl::col_traits< Col >, 53
 cl::data_set_accessor< Chan >, 66
 cl::DataPoint, 67
 channel, 68
 DataPoint, 68
 fileName, 68
 operator<<, 70
 sensor, 69
 time, 69
 value, 70
 cl::DataSet, 71
 accelerometerAverage, 72
 accelerometerMaximum, 72
 accelerometerX, 73
 accelerometerY, 73
 accelerometerZ, 74
 ChannelAccessor, 72
 create, 74
 extractId, 76
 fileName, 76
 gyroscopeAverage, 77
 gyroscopeMaximum, 77
 gyroscopeX, 78
 gyroscopeY, 78
 gyroscopeZ, 79
 hardwareTimestamp, 79
 rowCount, 80
 size_type, 72
 time, 80
 trigger, 81
 cl::Error, 81
 CL_ERROR_KIND, 82
 Error, 82
 file, 83
 function, 83
 Kind, 82
 kind, 83
 line, 84
 message, 84
 operator<<, 85
 raise, 84
 to_string, 85
 cl::Exception, 85
 base_type, 86
 Exception, 87
 file, 87
 function, 87
 line, 87
 cl::fs, 24
 directoryListing, 25
 DirectoryListingOption, 25
 ExcludeDotAndDotDot, 25
 formatError, 26
 None, 25
 operator<, 27
 operator<<, 27
 operator==, 27
 utf16ToUtf8, 28
 utf8ToUtf16, 29
 cl::fs::File, 88
 copyTo, 89
 create, 90
 exists, 90
 File, 88
 moveTo, 91
 path, 92
 remove, 92
 size, 93
 cl::fs::FileStream, 94
 ~FileStream, 96
 create, 96
 FileStream, 96
 OpenMode, 95
 operator=, 97
 PL_NONCOPYABLE, 98
 Read, 96
 readAll, 98
 ReadWrite, 96
 this_type, 95
 Write, 96
 write, 98
 cl::fs::Path, 113
 exists, 114
 isDirectory, 114
 isFile, 115
 operator<, 117
 operator<<, 117
 operator==, 117
 Path, 113
 str, 116
 CL_CHANNEL
 channel.hpp, 181
 cl, 13
 CL_CHANNEL_X
 above_threshold.cpp, 174

channel.cpp, 203
channel.hpp, 182
cl, 13
CL_ERROR_KIND
 cl::Error, 82
 error.hpp, 188
CL_ERROR_KIND_X
 error.cpp, 208
 error.hpp, 189
CL_FS_SEPARATOR
 separator.hpp, 195
CL_SENSOR
 cl, 14
 sensor.hpp, 200
CL_SENSOR_X
 cl, 14
 delete_non_bosch_sensors.cpp, 245
 sensor.cpp, 214
 sensor.hpp, 200
CL_SPECIALIZE_COL_TRAITS
 cl, 14–16
 column.hpp, 184
CL_THROW
 exception.hpp, 190
CL_THROW_FMT
 exception.hpp, 190
CL_UNEXPECTED
 error.hpp, 189
cm, 30
 Andre_1, 31
 Andre_2, 31
 Andre_3, 31
 Andre_Squats_1, 31
 Andre_Squats_2, 31
 DataSetIdentifier, 30
 Felix_11_17_39, 30
 Felix_12_50_00, 30
 Felix_13_00_09, 31
 Jan_1, 31
 Jan_2, 31
 Jan_3, 31
 Lucas_1, 31
 Lucas_2, 31
 Lucas_3, 31
 Marsi_14_59_59, 31
 Marsi_15_13_22, 31
 Marsi_15_31_36, 31
 Mike_14_07_33, 31
 Mike_14_14_32, 31
 Mike_14_20_28, 31
cm::ManualSegmentationPoint, 109
 asMilliseconds, 110
 frame, 111
 hour, 111
 ManualSegmentationPoint, 110
 minute, 111
 readCsvFile, 112
 second, 112
CMakeLists.txt
 include, 119–123
 set, 119–123
Column
 cl, 13
column.hpp
 CL_SPECIALIZE_COL_TRAITS, 184
column_index
 cl, 23
column_test.cpp
 TEST, 217
column_type
 cl, 12
compare_segmentation/CMakeLists.txt, 119
compare_segmentation/include/csv_line.hpp, 123
compare_segmentation/include/data_set_info.hpp, 124
compare_segmentation/include/filter_kind.hpp, 126
compare_segmentation/include/log_files.hpp, 127
compare_segmentation/include/log_info.hpp, 128
compare_segmentation/include/log_line.hpp, 129
compare_segmentation/include/paths.hpp, 129
compare_segmentation/include/segmentation_kind.hpp, 130
compare_segmentation/src/csv_line.cpp, 132
compare_segmentation/src/data_set_info.cpp, 132
compare_segmentation/src/filter_kind.cpp, 133
compare_segmentation/src/log_files.cpp, 133
compare_segmentation/src/log_info.cpp, 134
compare_segmentation/src/log_line.cpp, 135
compare_segmentation/src/main.cpp, 135
compare_segmentation/src/segmentation_kind.cpp, 146
compare_segmentation/test/CMakeLists.txt, 119
compare_segmentation/test/csv_line_test.cpp, 147
compare_segmentation/test/data_set_info_test.cpp, 147
compare_segmentation/test/log_files_test.cpp, 149
compare_segmentation/test/log_info_test.cpp, 150
compare_segmentation/test/log_line_test.cpp, 160
compare_segmentation/test/main.cpp, 137
confusion_matrix/CMakeLists.txt, 122
confusion_matrix/include/data_set_identifier.hpp, 163
confusion_matrix/include/manual_segmentation_point.hpp, 163
confusion_matrix/src/main.cpp, 144
confusion_matrix/src/manual_segmentation_point.cpp, 164
confusion_matrix/test/CMakeLists.txt, 123
confusion_matrix/test/main.cpp, 145
confusion_matrix/test/manual_segmentation_point_test.cpp, 165
convertToUnixLineEndings
 fmc, 48
copyTo
 cl::fs::File, 89
counting/CMakeLists.txt, 120
counting/include/above_threshold.hpp, 168

counting/include/average_comparison_value_calculator.hpp
 cs::LogInfo, 99
 create, 100
 counting/include/half_maximum_comparison_value_calculator.hpp
 deleteLowVariance, 101
 deleteTooClose, 101
 filterKind, 101
 invalidSensor, 105
 isInitialized, 102
 logFilePath, 102
 LogInfo, 100
 operator!=, 104
 counting/src/average_comparison_value_calculator.cpp, 175
 operator<<, 104
 operator==, 105
 segmentationKind, 102
 sensor, 103
 skipWindow, 103
 windowSize, 103
 counting/src/main.cpp, 138
 counting/src/run_above_threshold.cpp, 176
 counting/test/above_threshold_test.cpp, 177
 counting/test/CMakeLists.txt, 120
 counting/test/main.cpp, 140
 counting/test/percentage_of_test.cpp, 179
 create
 cl::DataSet, 74
 cl::fs::File, 90
 cl::fs::FileStream, 96
 cs::LogInfo, 100
 createBackupFile
 fmc, 49
 cs, 31
 Both, 33
 Butterworth, 33
 CS_SPECIALIZE_DATA_SET_INFO, 33–36
 FilterKind, 32
 logFiles, 36
 logPath, 42
 Maxima, 33
 Minima, 33
 MovingAverage, 33
 oldLogPath, 42
 operator!=, 38
 operator<<, 39
 operator==, 40
 PL_DEFINE_EXCEPTION_TYPE, 40
 repetitionCount, 40
 SegmentationKind, 33
 cs::CsvLineBuilder, 53
 build, 55
 CsvLineBuilder, 55
 dataSet, 55
 deleteLowVariance, 56
 deleteTooClose, 57
 filter, 58
 isOld, 59
 kind, 60
 repetitions, 61
 segmentationPoints, 62
 sensor, 63
 skipWindow, 64
 this_type, 54
 windowSize, 65
 cs::data_set_info< Tag >, 67
 cs::LogLine, 105
 fileName, 106
 filePath, 107
 invalidSensor, 109
 parse, 107
 segmentationPointCount, 108
 sensor, 108
 CS_SPECIALIZE_DATA_SET_INFO
 cs, 33–36
 data_set_info.hpp, 126
 csv_lib/CMakeLists.txt, 121
 csv_lib/include/cl/channel.hpp, 180
 csv_lib/include/cl/column.hpp, 183
 csv_lib/include/cl/data_point.hpp, 185
 csv_lib/include/cl/data_set.hpp, 186
 csv_lib/include/cl/dos2unix.hpp, 187
 csv_lib/include/cl/error.hpp, 188
 csv_lib/include/cl/exception.hpp, 189
 csv_lib/include/cl/fs/directory_listing.hpp, 191
 csv_lib/include/cl/fs/file.hpp, 192
 csv_lib/include/cl/fs/file_stream.hpp, 193
 csv_lib/include/cl/fs/path.hpp, 194
 csv_lib/include/cl/fs/separator.hpp, 194
 csv_lib/include/cl/fs/windows.hpp, 196
 csv_lib/include/cl/read_csv_file.hpp, 197
 csv_lib/include/cl/s2n.hpp, 198
 csv_lib/include/cl/sensor.hpp, 198
 csv_lib/include/cl/to_string.hpp, 201
 csv_lib/include/cl/use_unbuffered_io.hpp, 202
 csv_lib/src/cl/channel.cpp, 202
 csv_lib/src/cl/data_point.cpp, 204
 csv_lib/src/cl/data_set.cpp, 207
 csv_lib/src/cl/dos2unix.cpp, 207
 csv_lib/src/cl/error.cpp, 208
 csv_lib/src/cl/exception.cpp, 209
 csv_lib/src/cl/fs/directory_listing.cpp, 209
 csv_lib/src/cl/fs/file.cpp, 210
 csv_lib/src/cl/fs/file_stream.cpp, 210
 csv_lib/src/cl/fs/path.cpp, 211
 csv_lib/src/cl/fs/windows.cpp, 212
 csv_lib/src/cl/read_csv_file.cpp, 212
 csv_lib/src/cl/sensor.cpp, 213

csv_lib/src/cl/use_unbuffered_io.cpp, 214
csv_lib/test/channel_test.cpp, 215
csv_lib/test/CMakeLists.txt, 121
csv_lib/test/column_test.cpp, 217
csv_lib/test/data_point_test.cpp, 218
csv_lib/test/data_set_test.cpp, 220
csv_lib/test/directory_listing_test.cpp, 224
csv_lib/test/error_test.cpp, 226
csv_lib/test/exception_test.cpp, 228
csv_lib/test/main.cpp, 141
csv_lib/test/read_csv_file_test.cpp, 229
csv_lib/test/s2n_test.cpp, 230
csv_lib/test/sensor_test.cpp, 232
csv_lib/test/to_string_test.cpp, 233
csv_line_test.cpp
 TEST, 147
CsvFileKind
 cl, 13
CsvLineBuilder
 cs::CsvLineBuilder, 55
ctg, 43
 aboveThreshold, 43
 averageComparisonValueCalculator, 44
 halfMaximumComparisonValueCalculator, 44
 isRelevant, 45
 percentageOf, 46
 runAboveThreshold, 46

data_point.cpp
 channel, 204
 fileName, 204
 sensor, 205
 time, 205
 value, 206
data_point_test.cpp
 dp, 219
 TEST, 218, 219
data_set_accessor_v
 cl, 23
data_set_info.hpp
 CS_SPECIALIZE_DATA_SET_INFO, 126
data_set_info_test.cpp
 TEST, 148
data_set_test.cpp
 EXPECT_LONG_DOUBLE_EQ, 220
 TEST, 220–223
DataPoint
 cl::DataPoint, 68
dataSet
 cs::CsvLineBuilder, 55
dataSetAccessor
 cl, 16
DataSetIdentifier
 cm, 30
delete_non_bosch_sensors.cpp
 CL_SENSOR_X, 245
deleteLowVariance
 cs::CsvLineBuilder, 56
 cs::LogInfo, 101
deleteNonBoschSensors
 fmc, 49
deleteOutOfBoundsValues
 fmc, 50
deleteTooClose
 cs::CsvLineBuilder, 57
 cs::LogInfo, 101
directory_listing_test.cpp
 TEST, 225
directoryListing
 cl::fs, 25
DirectoryListingOption
 cl::fs, 25
dos2unix
 cl, 16
dp
 data_point_test.cpp, 219
DSI
 manual_segmentation_point.cpp, 165

Error
 cl::Error, 82
error
 error_test.cpp, 227
error.cpp
 CL_ERROR_KIND_X, 208
error.hpp
 CL_ERROR_KIND, 188
 CL_ERROR_KIND_X, 189
 CL_UNEXPECTED, 189
error_test.cpp
 error, 227
 TEST, 227
Exception
 cl::Exception, 87
exception.hpp
 CL_THROW, 190
 CL_THROW_FMT, 190
exception_test.cpp
 TEST, 228
ExcludeDotAndDotDot
 cl::fs, 25
exists
 cl::fs::File, 90
 cl::fs::Path, 114
EXPECT_LONG_DOUBLE_EQ
 above_threshold_test.cpp, 177
 data_set_test.cpp, 220
 percentage_of_test.cpp, 179
Expected
 cl, 12
ExtractId
 cl, 13
extractId
 cl::DataSet, 76

Felix_11_17_39
 cm, 30
Felix_12_50_00

cm, 30
Felix_13_00_09
 cm, 31
File
 cl::fs::File, 88
file
 cl::Error, 83
 cl::Exception, 87
fileName
 cl::DataPoint, 68
 cl::DataSet, 76
 cs::LogLine, 106
 data_point.cpp, 204
filePath
 cs::LogLine, 107
FileStream
 cl::fs::FileStream, 96
filter
 cs::CsvLineBuilder, 58
FilterKind
 cs, 32
filterKind
 cs::LogInfo, 101
fix_csv/CMakeLists.txt, 122
fix_csv/include/adjust_hardware_timestamp.hpp, 234
fix_csv/include/convert_to_unix_line_endings.hpp, 235
fix_csv/include/create_backup_file.hpp, 236
fix_csv/include/delete_non_bosch_sensors.hpp, 237
fix_csv/include/delete_out_of_bounds_values.hpp, 238
fix_csv/include/remove_zeros_from_field.hpp, 239
fix_csv/include/restore_from_backup.hpp, 240
fix_csv/include/write_file.hpp, 241
fix_csv/src/adjust_hardware_timestamp.cpp, 242
fix_csv/src/convert_to_unix_line_endings.cpp, 243
fix_csv/src/create_backup_file.cpp, 244
fix_csv/src/delete_non_bosch_sensors.cpp, 244
fix_csv/src/delete_out_of_bounds_values.cpp, 245
fix_csv/src/main.cpp, 142
fix_csv/src/remove_zeros_from_field.cpp, 246
fix_csv/src/restore_from_backup.cpp, 246
fix_csv/src/write_file.cpp, 247
fix_csv/test/adjust_hardware_timestamp_test.cpp, 248
fix_csv/test/CMakeLists.txt, 122
fix_csv/test/main.cpp, 143
fix_csv/test/remove_zeros_from_field_test.cpp, 250
Fixed
 cl, 14
fmc, 47
 adjustHardwareTimestamp, 48
 convertToUnixLineEndings, 48
 createBackupFile, 49
 deleteNonBoschSensors, 49
 deleteOutOfBoundsValues, 50
 removeZerosFromField, 50
 restoreFromBackup, 51
 writeFile, 51
formatError
 cl::fs, 26
frame
 cm::ManualSegmentationPoint, 111
function
 cl::Error, 83
 cl::Exception, 87
gyroscopeAverage
 cl::DataSet, 77
gyroscopeMaximum
 cl::DataSet, 77
gyroscopeThreshold
 cl, 23
GyroscopeX
 cl, 13
gyroscopeX
 cl::DataSet, 78
GyroscopeY
 cl, 13
gyroscopeY
 cl::DataSet, 78
GyroscopeZ
 cl, 13
gyroscopeZ
 cl::DataSet, 79
halfMaximumComparisonValueCalculator
 ctg, 44
HardwareTimestamp
 cl, 13
hardwareTimestamp
 cl::DataSet, 79
hour
 cm::ManualSegmentationPoint, 111
include
 CMakeLists.txt, 119–123
invalidSensor
 cs::LogInfo, 105
 cs::LogLine, 109
isAccelerometer
 cl, 17
isDirectory
 cl::fs::Path, 114
isFile
 cl::fs::Path, 115
isGyroscope
 cl, 17
isInitialized
 cs::LogInfo, 102
isOld
 cs::CsvLineBuilder, 59
isRelevant
 ctg, 45
Jan_1
 cm, 31
Jan_2
 cm, 31
Jan_3

cm, 31
Kind
 cl::Error, 82
kind
 cl::Error, 83
 cs::CsvLineBuilder, 60

line
 cl::Error, 84
 cl::Exception, 87
log_files_test.cpp
 TEST, 149, 150
log_info_test.cpp
 TEST, 151–160
log_line_test.cpp
 TEST, 161, 162
logFilePath
 cs::LogInfo, 102
logFiles
 cs, 36
LogInfo
 cs::LogInfo, 100
logPath
 cs, 42
Lucas_1
 cm, 31
Lucas_2
 cm, 31
Lucas_3
 cm, 31

main
 main.cpp, 136, 138, 139, 141–145
main.cpp
 main, 136, 138, 139, 141–145
manual_segmentation_point.cpp
 DSI, 165
manual_segmentation_point_test.cpp
 TEST, 166, 167
ManualSegmentationPoint
 cm::ManualSegmentationPoint, 110
Marsi_14_59_59
 cm, 31
Marsi_15_13_22
 cm, 31
Marsi_15_31_36
 cm, 31
Maxima
 cs, 33
message
 cl::Error, 84
Mike_14_07_33
 cm, 31
Mike_14_14_32
 cm, 31
Mike_14_20_28
 cm, 31
Minima
 cs, 33
minute
 cm::ManualSegmentationPoint, 111
moveTo
 cl::fs::File, 91
MovingAverage
 cs, 33

None
 cl::fs, 25

oldLogPath
 cs, 42
OpenMode
 cl::fs::FileStream, 95
operator!=
 cs, 38
 cs::LogInfo, 104
operator<
 cl::fs, 27
 cl::fs::Path, 117
operator<<
 cl, 18, 19
 cl::DataPoint, 70
 cl::Error, 85
 cl::fs, 27
 cl::fs::Path, 117
 cs, 39
 cs::LogInfo, 104
operator=
 cl::fs::FileStream, 97
operator==
 cl::fs, 27
 cl::fs::Path, 117
 cs, 40
 cs::LogInfo, 105

parse
 cs::LogLine, 107
Path
 cl::fs::Path, 113
path
 cl::fs::File, 92
percentage_of_test.cpp
 EXPECT_LONG_DOUBLE_EQ, 179
 TEST, 179
percentageOf
 ctg, 46
PL_DEFINE_EXCEPTION_TYPE
 cs, 40
PL_NONCOPYABLE
 cl::fs::FileStream, 98

raise
 cl::Error, 84
Raw
 cl, 14
Read
 cl::fs::FileStream, 96

read_csv_file_test.cpp
 TEST, 229, 230
 readAll
 cl::fs::FileStream, 98
 readCsvFile
 cl, 20
 cm::ManualSegmentationPoint, 112
 ReadWrite
 cl::fs::FileStream, 96
 remove
 cl::fs::File, 92
 remove_zeros_from_field_test.cpp
 TEST, 251–253
 removeZerosFromField
 fmc, 50
 repetitionCount
 cs, 40
 repetitions
 cs::CsvLineBuilder, 61
 restoreFromBackup
 fmc, 51
 rowCount
 cl::DataSet, 80
 runAboveThreshold
 ctg, 46
 s2n
 cl, 20
 s2n_test.cpp
 TEST, 231
 SamplingRate
 cl, 13
 second
 cm::ManualSegmentationPoint, 112
 SegmentationKind
 cs, 33
 segmentationKind
 cs::LogInfo, 102
 segmentationPointCount
 cs::LogLine, 108
 segmentationPoints
 cs::CsvLineBuilder, 62
 Sensor
 cl, 14
 sensor
 cl::DataPoint, 69
 cs::CsvLineBuilder, 63
 cs::LogInfo, 103
 cs::LogLine, 108
 data_point.cpp, 205
 sensor.cpp
 CL_SENSOR_X, 214
 sensor.hpp
 CL_SENSOR, 200
 CL_SENSOR_X, 200
 sensor_test.cpp
 TEST, 233
 sensors
 cl, 24
 separator.hpp
 CL_FS_SEPARATOR, 195
 set
 CMakeLists.txt, 119–123
 size
 cl::fs::File, 93
 size_type
 cl::DataSet, 72
 skipWindow
 cs::CsvLineBuilder, 64
 cs::LogInfo, 103
 str
 cl::fs::Path, 116
 TEST
 above_threshold_test.cpp, 178
 adjust_hardware_timestamp_test.cpp, 248–250
 channel_test.cpp, 215, 216
 column_test.cpp, 217
 csv_line_test.cpp, 147
 data_point_test.cpp, 218, 219
 data_set_info_test.cpp, 148
 data_set_test.cpp, 220–223
 directory_listing_test.cpp, 225
 error_test.cpp, 227
 exception_test.cpp, 228
 log_files_test.cpp, 149, 150
 log_info_test.cpp, 151–160
 log_line_test.cpp, 161, 162
 manual_segmentation_point_test.cpp, 166, 167
 percentage_of_test.cpp, 179
 read_csv_file_test.cpp, 229, 230
 remove_zeros_from_field_test.cpp, 251–253
 s2n_test.cpp, 231
 sensor_test.cpp, 233
 to_string_test.cpp, 234
 this_type
 cl::fs::FileStream, 95
 cs::CsvLineBuilder, 54
 threshold
 cl, 21
 Time
 cl, 13
 time
 cl::DataPoint, 69
 cl::DataSet, 80
 data_point.cpp, 205
 to_string
 cl, 21
 cl::Error, 85
 to_string_test.cpp
 TEST, 234
 Trigger
 cl, 13
 trigger
 cl::DataSet, 81
 useUnbufferedIo
 cl, 22

utf16ToUtf8
 cl::fs, [28](#)
utf8ToUtf16
 cl::fs, [29](#)

value
 cl::DataPoint, [70](#)
 data_point.cpp, [206](#)

windowSize
 cs::CsvLineBuilder, [65](#)
 cs::LogInfo, [103](#)

Write
 cl::fs::FileStream, [96](#)

write
 cl::fs::FileStream, [98](#)

writeFile
 fmc, [51](#)