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==() . . . 27
5.2.2.6 utf16ToUtf8() . . . 28
5.2.2.7 utf8ToUtf16() . . . 29
5.3 cm Namespace Reference . . . 29
5.3.1 Enumeration Type Documentation . . . 30
5.3.1.1 DataSetIdentifier . . . 30
5.3.1.2 Imu . . . 31
5.3.2 Function Documentation . . . 31
5.3.2.1 createSegmentationResults() . . . 31
5.3.2.2 interpolatedDataSetPaths() . . . 32
5.3.2.3 operator"!=() [1/2] . . . 33
5.3.2.4 operator"!=() [2/2] . . . 34
5.3.2.5 operator<<() [1/4] . . . 34
5.3.2.6 operator<<() [2/4] . . . 34
5.3.2.7 operator<<() [3/4] . . . 35
5.3.2.8 operator<<() [4/4] . . . 36
5.3.2.9 operator==() [1/2] . . . 36
5.3.2.10 operator==() [2/2] . . . 36
5.3.2.11 pythonOutput() . . . 37
5.3.2.12 segment() . . . 38
5.3.2.13 splitString() . . . 40
5.3.2.14 toDataSetIdentifier() . . . 40
5.3.3 Variable Documentation . . . 41

5.3.3.1 imuCount . . . 41
5.3.3.2 imus . . . 42
5.4 cs Namespace Reference . . . 42
5.4.1 Enumeration Type Documentation . . . 43
5.4.1.1 FilterKind . . . 43
5.4.1.2 SegmentationKind . . . 43
5.4.2 Function Documentation . . . 44
5.4.2.1 CS_SPECIALIZE_DATA_SET_INFO() [1/20] . . . 44
5.4.2.2 CS_SPECIALIZE_DATA_SET_INFO() [2/20] . . . 44
5.4.2.3 CS_SPECIALIZE_DATA_SET_INFO() [3/20] . . . 44
5.4.2.4 CS_SPECIALIZE_DATA_SET_INFO() [4/20] . . . 44
5.4.2.5 CS_SPECIALIZE_DATA_SET_INFO() [5/20] . . . 45
5.4.2.6 CS_SPECIALIZE_DATA_SET_INFO() [6/20] . . . 45
5.4.2.7 CS_SPECIALIZE_DATA_SET_INFO() [7/20] . . . 45
5.4.2.8 CS_SPECIALIZE_DATA_SET_INFO() [8/20] . . . 45
5.4.2.9 CS_SPECIALIZE_DATA_SET_INFO() [9/20] . . . 45
5.4.2.10 CS_SPECIALIZE_DATA_SET_INFO() [10/20] . . . 45
5.4.2.11 CS_SPECIALIZE_DATA_SET_INFO() [11/20] . . . 46
5.4.2.12 CS_SPECIALIZE_DATA_SET_INFO() [12/20] . . . 46
5.4.2.13 CS_SPECIALIZE_DATA_SET_INFO() [13/20] . . . 46
5.4.2.14 CS_SPECIALIZE_DATA_SET_INFO() [14/20] . . . 46
5.4.2.15 CS_SPECIALIZE_DATA_SET_INFO() [15/20] . . . 46
5.4.2.16 CS_SPECIALIZE_DATA_SET_INFO() [16/20] . . . 46
5.4.2.17 CS_SPECIALIZE_DATA_SET_INFO() [17/20] . . . 47
5.4.2.18 CS_SPECIALIZE_DATA_SET_INFO() [18/20] . . . 47
5.4.2.19 CS_SPECIALIZE_DATA_SET_INFO() [19/20] . . . 47
5.4.2.20 CS_SPECIALIZE_DATA_SET_INFO() [20/20] . . . 47
5.4.2.21 logFiles() . . . 47
5.4.2.22 operator"!=() . . . 48
5.4.2.23 operator<<() [1/3] . . . 49
5.4.2.24 operator<<() [2/3] . . . 49
5.4.2.25 operator<<() [3/3] . . . 49
5.4.2.26 operator==() . . . 50
5.4.2.27 PL_DEFINE_EXCEPTION_TYPE() . . . 50
5.4.2.28 repetitionCount() . . . 50
5.4.3 Variable Documentation . . . 51
5.4.3.1 logPath . . . 51
5.4.3.2 oldLogPath . . . 51
5.5 ctg Namespace Reference . . . 52
5.5.1 Function Documentation . . . 52
5.5.1.1 aboveThreshold() . . . 52
5.5.1.2 averageComparisonValueCalculator() . . . 53

5.5.1.3 halfMaximumComparisonValueCalculator() . . . 54
5.5.1.4 isRelevant() . . . 54
5.5.1.5 percentageOf() . . . 55
5.5.1.6 runAboveThreshold() . . . 56
5.6 fmc Namespace Reference . . . 56
5.6.1 Function Documentation . . . 57
5.6.1.1 adjustHardwareTimestamp() . . . 57
5.6.1.2 convertToUnixLineEndings() . . . 57
5.6.1.3 createBackupFile() . . . 58
5.6.1.4 deleteNonBoschSensors() . . . 59
5.6.1.5 deleteOutOfBoundsValues() . . . 59
5.6.1.6 removeZerosFromField() . . . 59
5.6.1.7 restoreFromBackup() . . . 60
5.6.1.8 writeFile() . . . 60

6 Class Documentation **63**
6.1 cm::Configuration::Builder Class Reference . . . 63
6.1.1 Detailed Description . . . 63
6.1.2 Constructor & Destructor Documentation . . . 64
6.1.2.1 Builder() . . . 64
6.1.3 Member Function Documentation . . . 64
6.1.3.1 build() . . . 64
6.1.3.2 deleteTooClose() . . . 65
6.1.3.3 deleteTooLowVariance() . . . 66
6.1.3.4 filterKind() . . . 67
6.1.3.5 imu() . . . 68
6.1.3.6 segmentationKind() . . . 69
6.1.3.7 skipWindow() . . . 70
6.1.3.8 windowSize() . . . 71
6.2 cl::col_traits< Col > Struct Template Reference . . . 72
6.2.1 Detailed Description . . . 72
6.3 cm::Configuration Class Reference . . . 73
6.3.1 Detailed Description . . . 74
6.3.2 Member Function Documentation . . . 74
6.3.2.1 deleteTooClose() . . . 74
6.3.2.2 deleteTooCloseOptions() . . . 75
6.3.2.3 deleteTooLowVariance() . . . 75
6.3.2.4 deleteTooLowVarianceOptions() . . . 76
6.3.2.5 filterKind() . . . 76
6.3.2.6 filterKindOptions() . . . 77
6.3.2.7 imu() . . . 77
6.3.2.8 imuOptions() . . . 78

6.3.2.9 segmentationKind() . . . 78
6.3.2.10 segmentationKindOptions() . . . 79
6.3.2.11 skipWindow() . . . 79
6.3.2.12 skipWindowOptions() . . . 80
6.3.2.13 windowSize() . . . 80
6.3.2.14 windowSizeOptions() . . . 81
6.3.3 Friends And Related Function Documentation . . . 81
6.3.3.1 Builder . . . 81
6.3.3.2 operator"!= . . . 81
6.3.3.3 operator<< . . . 82
6.3.3.4 operator== . . . 82
6.3.3.5 std::hash< Configuration > . . . 83
6.4 cs::CsvLineBuilder Class Reference . . . 83
6.4.1 Detailed Description . . . 84
6.4.2 Member Typedef Documentation . . . 84
6.4.2.1 this_type . . . 84
6.4.3 Constructor & Destructor Documentation . . . 84
6.4.3.1 CsvLineBuilder() . . . 84
6.4.4 Member Function Documentation . . . 84
6.4.4.1 build() . . . 85
6.4.4.2 dataSet() . . . 85
6.4.4.3 deleteLowVariance() . . . 86
6.4.4.4 deleteTooClose() . . . 87
6.4.4.5 filter() . . . 88
6.4.4.6 isOld() . . . 89
6.4.4.7 kind() . . . 90
6.4.4.8 repetitions() . . . 91
6.4.4.9 segmentationPoints() . . . 92
6.4.4.10 sensor() . . . 93
6.4.4.11 skipWindow() . . . 94
6.4.4.12 windowSize() . . . 95
6.5 cl::data_set_accessor< Chan > Struct Template Reference . . . 96
6.5.1 Detailed Description . . . 96
6.6 cs::data_set_info< Tag > Struct Template Reference . . . 97
6.6.1 Detailed Description . . . 97
6.7 cl::DataPoint Class Reference . . . 97
6.7.1 Detailed Description . . . 97
6.7.2 Constructor & Destructor Documentation . . . 98
6.7.2.1 DataPoint() . . . 98
6.7.3 Member Function Documentation . . . 98
6.7.3.1 channel() . . . 98
6.7.3.2 fileName() . . . 99

6.7.3.3 sensor() . . . 99
6.7.3.4 time() . . . 100
6.7.3.5 value() . . . 100
6.7.4 Friends And Related Function Documentation . . . 100
6.7.4.1 operator<< . . . 101
6.8 cl::DataSet Class Reference . . . 101
6.8.1 Detailed Description . . . 102
6.8.2 Member Typedef Documentation . . . 102
6.8.2.1 ChannelAccessor . . . 102
6.8.2.2 size_type . . . 102
6.8.3 Member Function Documentation . . . 102
6.8.3.1 accelerometerAverage() . . . 102
6.8.3.2 accelerometerMaximum() . . . 103
6.8.3.3 accelerometerX() . . . 103
6.8.3.4 accelerometerY() . . . 104
6.8.3.5 accelerometerZ() . . . 104
6.8.3.6 create() . . . 105
6.8.3.7 extractId() . . . 106
6.8.3.8 fileName() . . . 106
6.8.3.9 gyroscopeAverage() . . . 107
6.8.3.10 gyroscopeMaximum() . . . 108
6.8.3.11 gyroscopeX() . . . 108
6.8.3.12 gyroscopeY() . . . 109
6.8.3.13 gyroscopeZ() . . . 109
6.8.3.14 hardwareTimestamp() . . . 110
6.8.3.15 rowCount() . . . 110
6.8.3.16 time() . . . 110
6.8.3.17 trigger() . . . 111
6.9 cl::Error Class Reference . . . 111
6.9.1 Detailed Description . . . 112
6.9.2 Member Enumeration Documentation . . . 112
6.9.2.1 Kind . . . 112
6.9.3 Constructor & Destructor Documentation . . . 112
6.9.3.1 Error() . . . 112
6.9.4 Member Function Documentation . . . 113
6.9.4.1 file() . . . 113
6.9.4.2 function() . . . 113
6.9.4.3 kind() . . . 114
6.9.4.4 line() . . . 114
6.9.4.5 message() . . . 114
6.9.4.6 raise() . . . 115
6.9.4.7 to_string() . . . 115

6.9.5 Friends And Related Function Documentation . . . 115
6.9.5.1 operator<< . . . 115
6.10 cl::Exception Class Reference . . . 115
6.10.1 Detailed Description . . . 116
6.10.2 Member Typedef Documentation . . . 116
6.10.2.1 base_type . . . 116
6.10.3 Constructor & Destructor Documentation . . . 117
6.10.3.1 Exception() [1/2] . . . 117
6.10.3.2 Exception() [2/2] . . . 117
6.10.4 Member Function Documentation . . . 117
6.10.4.1 file() . . . 117
6.10.4.2 function() . . . 118
6.10.4.3 line() . . . 118
6.11 cl::fs::File Class Reference . . . 118
6.11.1 Detailed Description . . . 119
6.11.2 Constructor & Destructor Documentation . . . 119
6.11.2.1 File() . . . 119
6.11.3 Member Function Documentation . . . 120
6.11.3.1 copyTo() . . . 120
6.11.3.2 create() . . . 121
6.11.3.3 exists() . . . 121
6.11.3.4 moveTo() . . . 122
6.11.3.5 path() . . . 123
6.11.3.6 remove() . . . 123
6.11.3.7 size() . . . 124
6.12 cl::fs::FileStream Class Reference . . . 124
6.12.1 Detailed Description . . . 125
6.12.2 Member Typedef Documentation . . . 125
6.12.2.1 this_type . . . 125
6.12.3 Member Enumeration Documentation . . . 125
6.12.3.1 OpenMode . . . 125
6.12.4 Constructor & Destructor Documentation . . . 126
6.12.4.1 FileStream() . . . 126
6.12.4.2 ~FileStream() . . . 126
6.12.5 Member Function Documentation . . . 126
6.12.5.1 create() . . . 126
6.12.5.2 operator=() . . . 127
6.12.5.3 PL_NONCOPYABLE() . . . 128
6.12.5.4 readAll() . . . 128
6.12.5.5 write() . . . 128
6.13 std::hash<::cl::fs::Path > Struct Reference . . . 129
6.13.1 Detailed Description . . . 129

6.13.2 Member Function Documentation . . . 129
6.13.2.1 operator()() . . . 129
6.14 std::hash<::cm::Configuration > Struct Reference . . . 129
6.14.1 Detailed Description . . . 129
6.14.2 Member Function Documentation . . . 129
6.14.2.1 operator()() . . . 130
6.15 cs::LogInfo Class Reference . . . 130
6.15.1 Detailed Description . . . 131
6.15.2 Constructor & Destructor Documentation . . . 131
6.15.2.1 LogInfo() . . . 131
6.15.3 Member Function Documentation . . . 131
6.15.3.1 create() . . . 131
6.15.3.2 deleteLowVariance() . . . 132
6.15.3.3 deleteTooClose() . . . 132
6.15.3.4 filterKind() . . . 133
6.15.3.5 isInitialized() . . . 133
6.15.3.6 logFilePath() . . . 133
6.15.3.7 segmentationKind() . . . 134
6.15.3.8 sensor() . . . 134
6.15.3.9 skipWindow() . . . 134
6.15.3.10 windowSize() . . . 135
6.15.4 Friends And Related Function Documentation . . . 135
6.15.4.1 operator"!= . . . 135
6.15.4.2 operator<< . . . 135
6.15.4.3 operator== . . . 136
6.15.5 Member Data Documentation . . . 136
6.15.5.1 invalidSensor . . . 136
6.16 cs::LogLine Class Reference . . . 136
6.16.1 Detailed Description . . . 137
6.16.2 Member Function Documentation . . . 137
6.16.2.1 fileName() . . . 137
6.16.2.2 filePath() . . . 138
6.16.2.3 parse() . . . 138
6.16.2.4 segmentationPointCount() . . . 139
6.16.2.5 sensor() . . . 139
6.16.3 Member Data Documentation . . . 140
6.16.3.1 invalidSensor . . . 140
6.17 cm::ManualSegmentationPoint Class Reference . . . 140
6.17.1 Detailed Description . . . 141
6.17.2 Constructor & Destructor Documentation . . . 141
6.17.2.1 ManualSegmentationPoint() . . . 141
6.17.3 Member Function Documentation . . . 142

6.17.3.1 asMilliseconds() . . . 142
6.17.3.2 frame() . . . 142
6.17.3.3 hour() . . . 143
6.17.3.4 minute() . . . 143
6.17.3.5 readCsvFile() . . . 144
6.17.3.6 second() . . . 144
6.17.4 Friends And Related Function Documentation . . . 145
6.17.4.1 operator"!= . . . 145
6.17.4.2 operator<< . . . 145
6.17.4.3 operator== . . . 146
6.18 cl::fs::Path Class Reference . . . 146
6.18.1 Detailed Description . . . 147
6.18.2 Constructor & Destructor Documentation . . . 147
6.18.2.1 Path() [1/2] . . . 147
6.18.2.2 Path() [2/2] . . . 147
6.18.3 Member Function Documentation . . . 148
6.18.3.1 exists() . . . 148
6.18.3.2 isDirectory() . . . 148
6.18.3.3 isFile() . . . 149
6.18.3.4 str() . . . 150
6.18.4 Friends And Related Function Documentation . . . 151
6.18.4.1 operator< . . . 151
6.18.4.2 operator<< . . . 151
6.18.4.3 operator== . . . 152
6.19 cl::Process Class Reference . . . 152
6.19.1 Detailed Description . . . 153
6.19.2 Member Typedef Documentation . . . 153
6.19.2.1 this_type . . . 153
6.19.3 Constructor & Destructor Documentation . . . 153
6.19.3.1 Process() . . . 153
6.19.3.2 ~Process() . . . 153
6.19.4 Member Function Documentation . . . 153
6.19.4.1 create() . . . 154
6.19.4.2 file() [1/2] . . . 154
6.19.4.3 file() [2/2] . . . 154
6.19.4.4 operator=() . . . 154
6.19.4.5 PL_NONCOPYABLE() . . . 154

7 File Documentation **155**
7.1 compare_segmentation/CMakeLists.txt File Reference . . . 155
7.1.1 Function Documentation . . . 155
7.1.1.1 set() . . . 155

7.2 compare_segmentation/test/CMakeLists.txt File Reference . . . 155
7.2.1 Function Documentation . . . 155
7.2.1.1 include() . . . 156
7.3 counting/CMakeLists.txt File Reference . . . 156
7.3.1 Function Documentation . . . 156
7.3.1.1 set() . . . 156
7.4 counting/test/CMakeLists.txt File Reference . . . 156
7.4.1 Function Documentation . . . 156
7.4.1.1 include() . . . 156
7.5 csv_lib/CMakeLists.txt File Reference . . . 157
7.5.1 Function Documentation . . . 157
7.5.1.1 set() . . . 157
7.6 csv_lib/test/CMakeLists.txt File Reference . . . 157
7.6.1 Function Documentation . . . 157
7.6.1.1 include() . . . 157
7.7 fix_csv/CMakeLists.txt File Reference . . . 158
7.7.1 Function Documentation . . . 158
7.7.1.1 set() . . . 158
7.8 fix_csv/test/CMakeLists.txt File Reference . . . 158
7.8.1 Function Documentation . . . 158
7.8.1.1 include() . . . 158
7.9 confusion_matrix/CMakeLists.txt File Reference . . . 159
7.9.1 Function Documentation . . . 159
7.9.1.1 set() . . . 159
7.10 confusion_matrix/test/CMakeLists.txt File Reference . . . 159
7.10.1 Function Documentation . . . 159
7.10.1.1 include() . . . 159
7.11 compare_segmentation/include/csv_line.hpp File Reference . . . 160
7.12 compare_segmentation/include/data_set_info.hpp File Reference . . . 161
7.12.1 Macro Definition Documentation . . . 162
7.12.1.1 CS_SPECIALIZE_DATA_SET_INFO . . . 162
7.13 compare_segmentation/include/filter_kind.hpp File Reference . . . 163
7.14 compare_segmentation/include/log_files.hpp File Reference . . . 164
7.15 compare_segmentation/include/log_info.hpp File Reference . . . 164
7.16 compare_segmentation/include/log_line.hpp File Reference . . . 165
7.17 compare_segmentation/include/paths.hpp File Reference . . . 166
7.18 compare_segmentation/include/segmentation_kind.hpp File Reference . . . 167
7.19 compare_segmentation/src/csv_line.cpp File Reference . . . 168
7.20 compare_segmentation/src/data_set_info.cpp File Reference . . . 169
7.21 compare_segmentation/src/filter_kind.cpp File Reference . . . 169
7.22 compare_segmentation/src/log_files.cpp File Reference . . . 170
7.23 compare_segmentation/src/log_info.cpp File Reference . . . 171

7.24 compare_segmentation/src/log_line.cpp File Reference . . . 172
7.25 compare_segmentation/src/main.cpp File Reference . . . 172
7.25.1 Function Documentation . . . 173
7.25.1.1 main() . . . 173
7.26 compare_segmentation/test/main.cpp File Reference . . . 174
7.26.1 Function Documentation . . . 175
7.26.1.1 main() . . . 175
7.27 counting/src/main.cpp File Reference . . . 175
7.27.1 Function Documentation . . . 176
7.27.1.1 main() . . . 176
7.28 counting/test/main.cpp File Reference . . . 177
7.28.1 Function Documentation . . . 178
7.28.1.1 main() . . . 178
7.29 csv_lib/test/main.cpp File Reference . . . 178
7.29.1 Function Documentation . . . 179
7.29.1.1 main() . . . 179
7.30 fix_csv/src/main.cpp File Reference . . . 179
7.30.1 Function Documentation . . . 180
7.30.1.1 main() . . . 180
7.31 fix_csv/test/main.cpp File Reference . . . 180
7.31.1 Function Documentation . . . 181
7.31.1.1 main() . . . 181
7.32 confusion_matrix/src/main.cpp File Reference . . . 181
7.32.1 Function Documentation . . . 181
7.32.1.1 main() . . . 182
7.33 confusion_matrix/test/main.cpp File Reference . . . 182
7.33.1 Function Documentation . . . 183
7.33.1.1 main() . . . 183
7.34 compare_segmentation/src/segmentation_kind.cpp File Reference . . . 183
7.35 compare_segmentation/test/csv_line_test.cpp File Reference . . . 184
7.35.1 Function Documentation . . . 185
7.35.1.1 TEST() . . . 185
7.36 compare_segmentation/test/data_set_info_test.cpp File Reference . . . 185
7.36.1 Function Documentation . . . 185
7.36.1.1 TEST() . . . 186
7.37 compare_segmentation/test/log_files_test.cpp File Reference . . . 186
7.37.1 Function Documentation . . . 186
7.37.1.1 TEST() [1/3] . . . 187
7.37.1.2 TEST() [2/3] . . . 187
7.37.1.3 TEST() [3/3] . . . 188
7.38 compare_segmentation/test/log_info_test.cpp File Reference . . . 188
7.38.1 Function Documentation . . . 189

7.38.1.1 TEST() [1/19] . . . 189
7.38.1.2 TEST() [2/19] . . . 189
7.38.1.3 TEST() [3/19] . . . 190
7.38.1.4 TEST() [4/19] . . . 190
7.38.1.5 TEST() [5/19] . . . 191
7.38.1.6 TEST() [6/19] . . . 191
7.38.1.7 TEST() [7/19] . . . 192
7.38.1.8 TEST() [8/19] . . . 192
7.38.1.9 TEST() [9/19] . . . 193
7.38.1.10 TEST() [10/19] . . . 193
7.38.1.11 TEST() [11/19] . . . 194
7.38.1.12 TEST() [12/19] . . . 194
7.38.1.13 TEST() [13/19] . . . 195
7.38.1.14 TEST() [14/19] . . . 195
7.38.1.15 TEST() [15/19] . . . 196
7.38.1.16 TEST() [16/19] . . . 196
7.38.1.17 TEST() [17/19] . . . 197
7.38.1.18 TEST() [18/19] . . . 197
7.38.1.19 TEST() [19/19] . . . 198
7.39 compare_segmentation/test/log_line_test.cpp File Reference . . . 198
7.39.1 Function Documentation . . . 198
7.39.1.1 TEST() [1/4] . . . 199
7.39.1.2 TEST() [2/4] . . . 199
7.39.1.3 TEST() [3/4] . . . 200
7.39.1.4 TEST() [4/4] . . . 200
7.40 confusion_matrix/include/configuration.hpp File Reference . . . 200
7.41 confusion_matrix/include/create_segmentation_results.hpp File Reference . . . 201
7.42 confusion_matrix/include/data_set_identifier.hpp File Reference . . . 202
7.42.1 Macro Definition Documentation . . . 204
7.42.1.1 CM_DATA_SET_IDENTIFIER . . . 204
7.42.1.2 CM_DATA_SET_IDENTIFIER_X . . . 204
7.43 confusion_matrix/include/imu.hpp File Reference . . . 204
7.43.1 Macro Definition Documentation . . . 205
7.43.1.1 CM_IMU . . . 206
7.43.1.2 CM_IMU_X [1/3] . . . 206
7.43.1.3 CM_IMU_X [2/3] . . . 206
7.43.1.4 CM_IMU_X [3/3] . . . 206
7.44 confusion_matrix/include/interpolated_data_set_paths.hpp File Reference . . . 207
7.45 confusion_matrix/include/manual_segmentation_point.hpp File Reference . . . 208
7.46 confusion_matrix/include/python_output.hpp File Reference . . . 209
7.47 confusion_matrix/include/segment.hpp File Reference . . . 209
7.48 confusion_matrix/include/split_string.hpp File Reference . . . 210

7.49 confusion_matrix/src/configuration.cpp File Reference . . . 211
7.49.1 Macro Definition Documentation . . . 212
7.49.1.1 CM_CONTAINS . . . 212
7.49.1.2 CM_ENSURE_CONTAINS . . . 212
7.49.1.3 CM_ENSURE_HAS_VALUE . . . 213
7.50 confusion_matrix/src/create_segmentation_results.cpp File Reference . . . 213
7.51 confusion_matrix/src/data_set_identifier.cpp File Reference . . . 213
7.51.1 Macro Definition Documentation . . . 214
7.51.1.1 CM_DATA_SET_IDENTIFIER_X . . . 214
7.51.1.2 DSI . . . 214
7.52 confusion_matrix/src/imu.cpp File Reference . . . 215
7.52.1 Macro Definition Documentation . . . 215
7.52.1.1 CM_IMU_X . . . 215
7.53 confusion_matrix/src/interpolated_data_set_paths.cpp File Reference . . . 216
7.54 confusion_matrix/src/manual_segmentation_point.cpp File Reference . . . 216
7.54.1 Macro Definition Documentation . . . 217
7.54.1.1 DSI . . . 217
7.55 confusion_matrix/src/python_output.cpp File Reference . . . 217
7.55.1 Macro Definition Documentation . . . 218
7.55.1.1 CM_DEV_NULL . . . 218
7.55.1.2 CM_SEGMENTOR . . . 218
7.56 confusion_matrix/src/segment.cpp File Reference . . . 218
7.57 confusion_matrix/src/split_string.cpp File Reference . . . 219
7.58 confusion_matrix/test/data_set_identifier_test.cpp File Reference . . . 219
7.58.1 Macro Definition Documentation . . . 220
7.58.1.1 DSI . . . 220
7.58.2 Function Documentation . . . 220
7.58.2.1 TEST() . . . 220
7.59 confusion_matrix/test/interpolated_data_set_paths_test.cpp File Reference . . . 221
7.59.1 Function Documentation . . . 221
7.59.1.1 TEST() . . . 221
7.60 confusion_matrix/test/manual_segmentation_point_test.cpp File Reference . . . 222
7.60.1 Macro Definition Documentation . . . 222
7.60.1.1 DSI . . . 222
7.60.2 Function Documentation . . . 223
7.60.2.1 TEST() [1/11] . . . 223
7.60.2.2 TEST() [2/11] . . . 223
7.60.2.3 TEST() [3/11] . . . 223
7.60.2.4 TEST() [4/11] . . . 224
7.60.2.5 TEST() [5/11] . . . 224
7.60.2.6 TEST() [6/11] . . . 224
7.60.2.7 TEST() [7/11] . . . 224

7.60.2.8 TEST() [8/11] . . . 224
7.60.2.9 TEST() [9/11] . . . 225
7.60.2.10 TEST() [10/11] . . . 225
7.60.2.11 TEST() [11/11] . . . 225
7.61 confusion_matrix/test/segment_test.cpp File Reference . . . 225
7.61.1 Macro Definition Documentation . . . 226
7.61.1.1 EXPECT_SEGMENTATION_POINTS . . . 226
7.61.2 Function Documentation . . . 226
7.61.2.1 TEST() . . . 226
7.62 confusion_matrix/test/split_string_test.cpp File Reference . . . 227
7.62.1 Function Documentation . . . 227
7.62.1.1 TEST() . . . 227
7.63 counting/include/above_threshold.hpp File Reference . . . 228
7.64 counting/include/average_comparison_value_calculator.hpp File Reference . . . 229
7.65 counting/include/half_maximum_comparison_value_calculator.hpp File Reference . . . 229
7.66 counting/include/is_relevant.hpp File Reference . . . 230
7.67 counting/include/percentage_of.hpp File Reference . . . 231
7.68 counting/include/run_above_threshold.hpp File Reference . . . 232
7.69 counting/src/above_threshold.cpp File Reference . . . 233
7.69.1 Macro Definition Documentation . . . 234
7.69.1.1 CL_CHANNEL_X . . . 234
7.69.2 Variable Documentation . . . 234
7.69.2.1 channel . . . 235
7.69.2.2 channelAccessor . . . 235
7.70 counting/src/average_comparison_value_calculator.cpp File Reference . . . 235
7.71 counting/src/half_maximum_comparison_value_calculator.cpp File Reference . . . 236
7.72 counting/src/run_above_threshold.cpp File Reference . . . 236
7.73 counting/test/above_threshold_test.cpp File Reference . . . 237
7.73.1 Macro Definition Documentation . . . 237
7.73.1.1 EXPECT_LONG_DOUBLE_EQ . . . 238
7.73.2 Function Documentation . . . 238
7.73.2.1 TEST() . . . 238
7.74 counting/test/percentage_of_test.cpp File Reference . . . 239
7.74.1 Macro Definition Documentation . . . 239
7.74.1.1 EXPECT_LONG_DOUBLE_EQ . . . 239
7.74.2 Function Documentation . . . 239
7.74.2.1 TEST() . . . 240
7.75 csv_lib/include/cl/channel.hpp File Reference . . . 240
7.75.1 Macro Definition Documentation . . . 241
7.75.1.1 CL_CHANNEL . . . 242
7.75.1.2 CL_CHANNEL_X [1/4] . . . 242
7.75.1.3 CL_CHANNEL_X [2/4] . . . 242

7.75.1.4 CL_CHANNEL_X [3/4] . . . 242
7.75.1.5 CL_CHANNEL_X [4/4] . . . 243
7.76 csv_lib/include/cl/column.hpp File Reference . . . 243
7.76.1 Macro Definition Documentation . . . 244
7.76.1.1 CL_SPECIALIZE_COL_TRAITS . . . 245
7.77 csv_lib/include/cl/data_point.hpp File Reference . . . 245
7.78 csv_lib/include/cl/data_set.hpp File Reference . . . 246
7.79 csv_lib/include/cl/dos2unix.hpp File Reference . . . 247
7.80 csv_lib/include/cl/error.hpp File Reference . . . 248
7.80.1 Macro Definition Documentation . . . 248
7.80.1.1 CL_ERROR_KIND . . . 249
7.80.1.2 CL_ERROR_KIND_X . . . 249
7.80.1.3 CL_UNEXPECTED . . . 249
7.81 csv_lib/include/cl/exception.hpp File Reference . . . 249
7.81.1 Macro Definition Documentation . . . 250
7.81.1.1 CL_THROW . . . 250
7.81.1.2 CL_THROW_FMT . . . 250
7.82 csv_lib/include/cl/fs/directory_listing.hpp File Reference . . . 251
7.83 csv_lib/include/cl/fs/file.hpp File Reference . . . 252
7.84 csv_lib/include/cl/fs/file_stream.hpp File Reference . . . 253
7.85 csv_lib/include/cl/fs/path.hpp File Reference . . . 254
7.86 csv_lib/include/cl/fs/separator.hpp File Reference . . . 254
7.86.1 Macro Definition Documentation . . . 255
7.86.1.1 CL_FS_SEPARATOR . . . 255
7.87 csv_lib/include/cl/fs/windows.hpp File Reference . . . 256
7.87.1 Detailed Description . . . 257
7.88 csv_lib/include/cl/process.hpp File Reference . . . 257
7.89 csv_lib/include/cl/read_csv_file.hpp File Reference . . . 258
7.90 csv_lib/include/cl/s2n.hpp File Reference . . . 259
7.91 csv_lib/include/cl/sensor.hpp File Reference . . . 259
7.91.1 Macro Definition Documentation . . . 261
7.91.1.1 CL_SENSOR . . . 261
7.91.1.2 CL_SENSOR_X [1/2] . . . 261
7.91.1.3 CL_SENSOR_X [2/2] . . . 261
7.92 csv_lib/include/cl/to_string.hpp File Reference . . . 262
7.93 csv_lib/include/cl/use_unbuffered_io.hpp File Reference . . . 263
7.94 csv_lib/src/cl/channel.cpp File Reference . . . 263
7.94.1 Macro Definition Documentation . . . 264
7.94.1.1 CL_CHANNEL_X [1/2] . . . 264
7.94.1.2 CL_CHANNEL_X [2/2] . . . 264
7.95 csv_lib/src/cl/data_point.cpp File Reference . . . 265
7.95.1 Function Documentation . . . 265

7.95.1.1 channel() . . . 265
7.95.1.2 fileName() . . . 266
7.95.1.3 sensor() . . . 266
7.95.1.4 time() . . . 267
7.95.1.5 value() . . . 267
7.96 csv_lib/src/cl/data_set.cpp File Reference . . . 268
7.97 csv_lib/src/cl/dos2unix.cpp File Reference . . . 268
7.98 csv_lib/src/cl/error.cpp File Reference . . . 269
7.98.1 Macro Definition Documentation . . . 269
7.98.1.1 CL_ERROR_KIND_X . . . 270
7.99 csv_lib/src/cl/exception.cpp File Reference . . . 270
7.100 csv_lib/src/cl/fs/directory_listing.cpp File Reference . . . 270
7.101 csv_lib/src/cl/fs/file.cpp File Reference . . . 271
7.102 csv_lib/src/cl/fs/file_stream.cpp File Reference . . . 271
7.103 csv_lib/src/cl/fs/path.cpp File Reference . . . 272
7.104 csv_lib/src/cl/fs/windows.cpp File Reference . . . 273
7.105 csv_lib/src/cl/process.cpp File Reference . . . 273
7.106 csv_lib/src/cl/read_csv_file.cpp File Reference . . . 274
7.107 csv_lib/src/cl/sensor.cpp File Reference . . . 275
7.107.1 Macro Definition Documentation . . . 275
7.107.1.1 CL_SENSOR_X . . . 275
7.108 csv_lib/src/cl/use_unbuffered_io.cpp File Reference . . . 276
7.109 csv_lib/test/channel_test.cpp File Reference . . . 276
7.109.1 Function Documentation . . . 277
7.109.1.1 TEST() [1/4] . . . 277
7.109.1.2 TEST() [2/4] . . . 277
7.109.1.3 TEST() [3/4] . . . 277
7.109.1.4 TEST() [4/4] . . . 278
7.110 csv_lib/test/column_test.cpp File Reference . . . 278
7.110.1 Function Documentation . . . 279
7.110.1.1 TEST() [1/2] . . . 279
7.110.1.2 TEST() [2/2] . . . 279
7.111 csv_lib/test/data_point_test.cpp File Reference . . . 280
7.111.1 Function Documentation . . . 280
7.111.1.1 TEST() [1/2] . . . 280
7.111.1.2 TEST() [2/2] . . . 281
7.111.2 Variable Documentation . . . 281
7.111.2.1 dp . . . 281
7.112 csv_lib/test/data_set_test.cpp File Reference . . . 282
7.112.1 Macro Definition Documentation . . . 282
7.112.1.1 EXPECT_LONG_DOUBLE_EQ . . . 282
7.112.2 Function Documentation . . . 282

7.112.2.1 TEST() [1/4] . . . 283
7.112.2.2 TEST() [2/4] . . . 283
7.112.2.3 TEST() [3/4] . . . 284
7.112.2.4 TEST() [4/4] . . . 285
7.113 csv_lib/test/directory_listing_test.cpp File Reference . . . 286
7.113.1 Function Documentation . . . 287
7.113.1.1 TEST() [1/3] . . . 287
7.113.1.2 TEST() [2/3] . . . 287
7.113.1.3 TEST() [3/3] . . . 288
7.114 csv_lib/test/error_test.cpp File Reference . . . 288
7.114.1 Function Documentation . . . 289
7.114.1.1 TEST() [1/4] . . . 289
7.114.1.2 TEST() [2/4] . . . 289
7.114.1.3 TEST() [3/4] . . . 289
7.114.1.4 TEST() [4/4] . . . 289
7.114.2 Variable Documentation . . . 289
7.114.2.1 error . . . 290
7.115 csv_lib/test/exception_test.cpp File Reference . . . 290
7.115.1 Function Documentation . . . 290
7.115.1.1 TEST() . . . 290
7.116 csv_lib/test/read_csv_file_test.cpp File Reference . . . 291
7.116.1 Function Documentation . . . 291
7.116.1.1 TEST() [1/2] . . . 291
7.116.1.2 TEST() [2/2] . . . 292
7.117 csv_lib/test/s2n_test.cpp File Reference . . . 292
7.117.1 Function Documentation . . . 293
7.117.1.1 TEST() [1/3] . . . 293
7.117.1.2 TEST() [2/3] . . . 293
7.117.1.3 TEST() [3/3] . . . 294
7.118 csv_lib/test/sensor_test.cpp File Reference . . . 294
7.118.1 Function Documentation . . . 295
7.118.1.1 TEST() [1/2] . . . 295
7.118.1.2 TEST() [2/2] . . . 295
7.119 csv_lib/test/to_string_test.cpp File Reference . . . 295
7.119.1 Function Documentation . . . 296
7.119.1.1 TEST() . . . 296
7.120 fix_csv/include/adjust_hardware_timestamp.hpp File Reference . . . 296
7.121 fix_csv/include/convert_to_unix_line_endings.hpp File Reference . . . 297
7.122 fix_csv/include/create_backup_file.hpp File Reference . . . 298
7.123 fix_csv/include/delete_non_bosch_sensors.hpp File Reference . . . 299
7.124 fix_csv/include/delete_out_of_bounds_values.hpp File Reference . . . 300
7.125 fix_csv/include/remove_zeros_from_field.hpp File Reference . . . 301

7.126 fix_csv/include/restore_from_backup.hpp File Reference 302
7.127 fix_csv/include/write_file.hpp File Reference 303
7.128 fix_csv/src/adjust_hardware_timestamp.cpp File Reference 304
7.129 fix_csv/src/convert_to_unix_line_endings.cpp File Reference 305
7.130 fix_csv/src/create_backup_file.cpp File Reference 306
7.131 fix_csv/src/delete_non_bosch_sensors.cpp File Reference 306
7.131.1 Macro Definition Documentation 307
7.131.1.1 CL_SENSOR_X 307
7.132 fix_csv/src/delete_out_of_bounds_values.cpp File Reference 307
7.133 fix_csv/src/remove_zeros_from_field.cpp File Reference 308
7.134 fix_csv/src/restore_from_backup.cpp File Reference 308
7.135 fix_csv/src/write_file.cpp File Reference 309
7.136 fix_csv/test/adjust_hardware_timestamp_test.cpp File Reference 310
7.136.1 Function Documentation 310
7.136.1.1 TEST() [1/5] 310
7.136.1.2 TEST() [2/5] 311
7.136.1.3 TEST() [3/5] 311
7.136.1.4 TEST() [4/5] 312
7.136.1.5 TEST() [5/5] 312
7.137 fix_csv/test/remove_zeros_from_field_test.cpp File Reference 312
7.137.1 Function Documentation 313
7.137.1.1 TEST() [1/6] 313
7.137.1.2 TEST() [2/6] 314
7.137.1.3 TEST() [3/6] 314
7.137.1.4 TEST() [4/6] 315
7.137.1.5 TEST() [5/6] 315
7.137.1.6 TEST() [6/6] 316

Index **317**

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

cl 11
cl::fs 24
cm 29
cs 42
ctg 52
fmc 56

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

cm::Configuration::Builder 63
cl::col_traits< Col > 72
cm::Configuration 73
cs::CsvLineBuilder 83
cl::data_set_accessor< Chan > 96
cs::data_set_info< Tag > 97
cl::DataPoint 97
cl::DataSet 101
cl::Error 111
std::exception
    std::runtime_error
        cl::Exception 115
cl::fs::File 118
cl::fs::FileStream 124
std::hash<::cl::fs::Path > 129
std::hash<::cm::Configuration > 129
cs::LogInfo 130
cs::LogLine 136
cm::ManualSegmentationPoint 140
cl::fs::Path 146
cl::Process 152

Chapter 3

Class Index

3.1 Class List

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

cm::Configuration::Builder
Builder type for `Configuration` 63
cl::col_traits< Col > 72
cm::Configuration
Represents a possible configuration for the Python segmentor 73
cs::CsvLineBuilder
Builder for a CSV line 83
cl::data_set_accessor< Chan > 96
cs::data_set_info< Tag >
Meta function for data set tags 97
cl::DataPoint 97
cl::DataSet 101
cl::Error 111
cl::Exception 115
cl::fs::File
Represents a file 118
cl::fs::FileStream
A binary file stream 124
std::hash<::cl::fs::Path > 129
std::hash<::cm::Configuration > 129
cs::LogInfo
Information about a log file 130
cs::LogLine
A line out of a log file 136
cm::ManualSegmentationPoint
Type used to represent a manual segmentation point 140
cl::fs::Path
A filesystem path 146
cl::Process 152

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

compare_segmentation/include/csv_line.hpp . . . 160
compare_segmentation/include/data_set_info.hpp . . . 161
compare_segmentation/include/filter_kind.hpp . . . 163
compare_segmentation/include/log_files.hpp . . . 164
compare_segmentation/include/log_info.hpp . . . 164
compare_segmentation/include/log_line.hpp . . . 165
compare_segmentation/include/paths.hpp . . . 166
compare_segmentation/include/segmentation_kind.hpp . . . 167
compare_segmentation/src/csv_line.cpp . . . 168
compare_segmentation/src/data_set_info.cpp . . . 169
compare_segmentation/src/filter_kind.cpp . . . 169
compare_segmentation/src/log_files.cpp . . . 170
compare_segmentation/src/log_info.cpp . . . 171
compare_segmentation/src/log_line.cpp . . . 172
compare_segmentation/src/main.cpp . . . 172
compare_segmentation/src/segmentation_kind.cpp . . . 183
compare_segmentation/test/csv_line_test.cpp . . . 184
compare_segmentation/test/data_set_info_test.cpp . . . 185
compare_segmentation/test/log_files_test.cpp . . . 186
compare_segmentation/test/log_info_test.cpp . . . 188
compare_segmentation/test/log_line_test.cpp . . . 198
compare_segmentation/test/main.cpp . . . 174
confusion_matrix/include/configuration.hpp . . . 200
confusion_matrix/include/create_segmentation_results.hpp . . . 201
confusion_matrix/include/data_set_identifier.hpp . . . 202
confusion_matrix/include/imu.hpp . . . 204
confusion_matrix/include/interpolated_data_set_paths.hpp . . . 207
confusion_matrix/include/manual_segmentation_point.hpp . . . 208
confusion_matrix/include/python_output.hpp . . . 209
confusion_matrix/include/segment.hpp . . . 209
confusion_matrix/include/split_string.hpp . . . 210
confusion_matrix/src/configuration.cpp . . . 211
confusion_matrix/src/create_segmentation_results.cpp . . . 213
confusion_matrix/src/data_set_identifier.cpp . . . 213
confusion_matrix/src/imu.cpp . . . 215

confusion_matrix/src/interpolated_data_set_paths.cpp . . . 216
confusion_matrix/src/main.cpp . . . 181
confusion_matrix/src/manual_segmentation_point.cpp . . . 216
confusion_matrix/src/python_output.cpp . . . 217
confusion_matrix/src/segment.cpp . . . 218
confusion_matrix/src/split_string.cpp . . . 219
confusion_matrix/test/data_set_identifier_test.cpp . . . 219
confusion_matrix/test/interpolated_data_set_paths_test.cpp . . . 221
confusion_matrix/test/main.cpp . . . 182
confusion_matrix/test/manual_segmentation_point_test.cpp . . . 222
confusion_matrix/test/segment_test.cpp . . . 225
confusion_matrix/test/split_string_test.cpp . . . 227
counting/include/above_threshold.hpp . . . 228
counting/include/average_comparison_value_calculator.hpp . . . 229
counting/include/half_maximum_comparison_value_calculator.hpp . . . 229
counting/include/is_relevant.hpp . . . 230
counting/include/percentage_of.hpp . . . 231
counting/include/run_above_threshold.hpp . . . 232
counting/src/above_threshold.cpp . . . 233
counting/src/average_comparison_value_calculator.cpp . . . 235
counting/src/half_maximum_comparison_value_calculator.cpp . . . 236
counting/src/main.cpp . . . 175
counting/src/run_above_threshold.cpp . . . 236
counting/test/above_threshold_test.cpp . . . 237
counting/test/main.cpp . . . 177
counting/test/percentage_of_test.cpp . . . 239
csv_lib/include/cl/channel.hpp . . . 240
csv_lib/include/cl/column.hpp . . . 243
csv_lib/include/cl/data_point.hpp . . . 245
csv_lib/include/cl/data_set.hpp . . . 246
csv_lib/include/cl/dos2unix.hpp . . . 247
csv_lib/include/cl/error.hpp . . . 248
csv_lib/include/cl/exception.hpp . . . 249
csv_lib/include/cl/process.hpp . . . 257
csv_lib/include/cl/read_csv_file.hpp . . . 258
csv_lib/include/cl/s2n.hpp . . . 259
csv_lib/include/cl/sensor.hpp . . . 259
csv_lib/include/cl/to_string.hpp . . . 262
csv_lib/include/cl/use_unbuffered_io.hpp . . . 263
csv_lib/include/cl/fs/directory_listing.hpp . . . 251
csv_lib/include/cl/fs/file.hpp . . . 252
csv_lib/include/cl/fs/file_stream.hpp . . . 253
csv_lib/include/cl/fs/path.hpp . . . 254
csv_lib/include/cl/fs/separator.hpp . . . 254
csv_lib/include/cl/fs/windows.hpp
Contains Microsoft Windows specific functions . . . 256
csv_lib/src/cl/channel.cpp . . . 263
csv_lib/src/cl/data_point.cpp . . . 265
csv_lib/src/cl/data_set.cpp . . . 268
csv_lib/src/cl/dos2unix.cpp . . . 268
csv_lib/src/cl/error.cpp . . . 269
csv_lib/src/cl/exception.cpp . . . 270
csv_lib/src/cl/process.cpp . . . 273
csv_lib/src/cl/read_csv_file.cpp . . . 274
csv_lib/src/cl/sensor.cpp . . . 275
csv_lib/src/cl/use_unbuffered_io.cpp . . . 276
csv_lib/src/cl/fs/directory_listing.cpp . . . 270
csv_lib/src/cl/fs/file.cpp . . . 271

csv_lib/src/cl/fs/file_stream.cpp . . . 271
csv_lib/src/cl/fs/path.cpp . . . 272
csv_lib/src/cl/fs/windows.cpp . . . 273
csv_lib/test/channel_test.cpp . . . 276
csv_lib/test/column_test.cpp . . . 278
csv_lib/test/data_point_test.cpp . . . 280
csv_lib/test/data_set_test.cpp . . . 282
csv_lib/test/directory_listing_test.cpp . . . 286
csv_lib/test/error_test.cpp . . . 288
csv_lib/test/exception_test.cpp . . . 290
csv_lib/test/main.cpp . . . 178
csv_lib/test/read_csv_file_test.cpp . . . 291
csv_lib/test/s2n_test.cpp . . . 292
csv_lib/test/sensor_test.cpp . . . 294
csv_lib/test/to_string_test.cpp . . . 295
fix_csv/include/adjust_hardware_timestamp.hpp . . . 296
fix_csv/include/convert_to_unix_line_endings.hpp . . . 297
fix_csv/include/create_backup_file.hpp . . . 298
fix_csv/include/delete_non_bosch_sensors.hpp . . . 299
fix_csv/include/delete_out_of_bounds_values.hpp . . . 300
fix_csv/include/remove_zeros_from_field.hpp . . . 301
fix_csv/include/restore_from_backup.hpp . . . 302
fix_csv/include/write_file.hpp . . . 303
fix_csv/src/adjust_hardware_timestamp.cpp . . . 304
fix_csv/src/convert_to_unix_line_endings.cpp . . . 305
fix_csv/src/create_backup_file.cpp . . . 306
fix_csv/src/delete_non_bosch_sensors.cpp . . . 306
fix_csv/src/delete_out_of_bounds_values.cpp . . . 307
fix_csv/src/main.cpp . . . 179
fix_csv/src/remove_zeros_from_field.cpp . . . 308
fix_csv/src/restore_from_backup.cpp . . . 308
fix_csv/src/write_file.cpp . . . 309
fix_csv/test/adjust_hardware_timestamp_test.cpp . . . 310
fix_csv/test/main.cpp . . . 180
fix_csv/test/remove_zeros_from_field_test.cpp . . . 312

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
- class Process

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 64 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

p	The beginning of the data to convert.
size	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-6 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

directoryPath	The directory to list.
directoryListingOption	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

errorCode	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

lhs	The left hand side operand.
rhs	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

os	the ostream to print to.
path	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

lhs	The left hand side operand.
rhs	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-6 string.

Parameters

utf16	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

utf8	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 Configuration

  Represents a possible configuration for the Python segmentor.
- class ManualSegmentationPoint

  Type used to represent a manual segmentation point.

Enumerations

- enum DataSetIdentifier { DataSetIdentifier::CM_DATA_SET_IDENTIFIER_X, DataSetIdentifier::CM_DATA_SET_IDENTIFIER }
- enum Imu { Imu::CM_IMU_X, Imu::CM_IMU }

Functions

- std::unordered_map< cm::Configuration, std::unordered_map< cl::fs::Path, std::vector< std::uint64_t > > > createSegmentationResults ()
- std::ostream & operator<< (std::ostream &os, DataSetIdentifier dsi)

  Prints a DataSetIdentifier to an ostream.
- DataSetIdentifier toDataSetIdentifier (const cl::fs::Path &path)

  Converts a path to a CSV file to the corresponding DataSetIdentifier.
- std::ostream & operator<< (std::ostream &os, Imu imu)
- std::vector< cl::fs::Path > interpolatedDataSetPaths ()

  Returns the paths to the interpolated data sets.
- std::string pythonOutput (const cl::fs::Path &csvFilePath, const Configuration &segmentorConfiguration)

  Runs the Python segmentor on `path`.
- std::unordered_map< cl::fs::Path, std::vector< std::uint64_t > > segment (const Configuration &segmentorConfiguration)
- std::vector< std::string > splitString (std::string string, pl::string_view splitBy)

  Splits `string` by `splitBy`.
- bool operator== (const Configuration &lhs, const Configuration &rhs) noexcept
- bool operator!= (const Configuration &lhs, const Configuration &rhs) noexcept
- std::ostream & operator<< (std::ostream &os, const Configuration &config)
- bool operator== (const ManualSegmentationPoint &lhs, const ManualSegmentationPoint &rhs) noexcept
- bool operator!= (const ManualSegmentationPoint &lhs, const ManualSegmentationPoint &rhs) noexcept
- std::ostream & operator<< (std::ostream &os, const ManualSegmentationPoint &manualSegmentationPoint)

Variables

- constexpr std::size_t imuCount
- constexpr std::array< Imu, imuCount > imus

5.3.1 Enumeration Type Documentation

5.3.1.1 DataSetIdentifier

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

Enumerator

CM_DATA_SET_IDENTIFIER↩ _X	
CM_DATA_SET_IDENTIFIER	

Definition at line 30 of file data_set_identifier.hpp.

5.3.1.2 Imu

```
enum cm::Imu [strong]
```

Enumerator

CM_IMU↩ _X	
CM_IMU	

Definition at line 14 of file imu.hpp.

5.3.2 Function Documentation

5.3.2.1 createSegmentationResults()

```
std::unordered_map< Configuration, std::unordered_map< cl::fs::Path, std::vector< std::uint64↩
_t > > > cm::createSegmentationResults ( )
```

Definition at line 8 of file create_segmentation_results.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

5.3.2.2 interpolatedDataSetPaths()

```
std::vector< cl::fs::Path > cm::interpolatedDataSetPaths ( )
```

Returns the paths to the interpolated data sets.

Returns

The interpolated data set paths.

Exceptions

cl::Exception	on error.

Definition at line 13 of file interpolated_data_set_paths.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

5.3.2.3 operator"!=() [1/2]

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

Parameters

lhs	The first operand.
rhs	The second operand.

Returns

true if `lhs` and `rhs` are considered not to be equal.

Definition at line 206 of file configuration.cpp.

5.3.2.4 operator"!=() [2/2]

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

Parameters

lhs	The first operand.
rhs	The second operand.

Returns

true if `lhs` is considered not equal to `rhs`; false otherwise.

Definition at line 139 of file manual_segmentation_point.cpp.

5.3.2.5 operator<<() [1/4]

```
std::ostream& cm::operator<< (
            std::ostream & os,
            const Configuration & config )
```

Parameters

os	The ostream to print to.
config	The `Configuration` to print.

Returns

`os`

Definition at line 211 of file configuration.cpp.

5.3.2.6 operator<<() [2/4]

```
std::ostream& cm::operator<< (
            std::ostream & os,
            const ManualSegmentationPoint & manualSegmentationPoint )
```

Parameters

os	The ostream to print to
manualSegmentationPoint	The `ManualSegmentationPoint` to print.

Returns

os

Definition at line 146 of file manual_segmentation_point.cpp.

Here is the call graph for this function:

5.3.2.7 operator<<() [3/4]

```
std::ostream & cm::operator<< (
            std::ostream & os,
            DataSetIdentifier dsi )
```

Prints a DataSetIdentifier to an ostream.

Parameters

os	The ostream to print to.
dsi	The `DataSetIdentifier` to print.

Returns

os

Definition at line 28 of file data_set_identifier.cpp.

5.3.2.8 operator<<() [4/4]

```
std::ostream & cm::operator<< (
            std::ostream & os,
            Imu imu )
```

Definition at line 25 of file imu.cpp.

5.3.2.9 operator==() [1/2]

```
bool cm::operator== (
            const Configuration & lhs,
            const Configuration & rhs )  [noexcept]
```

Parameters

lhs	The first operand.
rhs	The second operand.

Returns

true if `lhs` and `rhs` are considered to be equal.

Definition at line 186 of file configuration.cpp.

5.3.2.10 operator==() [2/2]

```
bool cm::operator== (
            const ManualSegmentationPoint & lhs,
            const ManualSegmentationPoint & rhs )  [noexcept]
```

Parameters

lhs	The first operand.
rhs	The second operand.

Returns

true if `lhs` is considered equal to `rhs`; false otherwise.

Definition at line 131 of file manual_segmentation_point.cpp.

5.3.2.11 pythonOutput()

```
std::string cm::pythonOutput (
            const cl::fs::Path & csvFilePath,
            const Configuration & segmentorConfiguration )
```

Runs the Python segmentor on `path`.

Parameters

path	The path to the CSV file to segment.
segmentorConfiguration	The configuration to use.

Returns

The output of the Python application.

Exceptions

cl::Exception	if creating the process failed.

Definition at line 22 of file python_output.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

5.3.2.12 segment()

```
std::unordered_map< cl::fs::Path, std::vector< std::uint64_t > > cm::segment (
            const Configuration & segmentorConfiguration )
```

Invokes Python to segment the interpolated data sets.

Parameters

segmentorConfiguration	The `Configuration` to use for the Python segmentor.

Returns

A map that maps the paths to the interpolated data sets to vectors of the hardware timestamps (in milliseconds) that are segmentation points.

Exceptions

cl::Exception	if an error occurs.

Definition at line 64 of file segment.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

5.3.2.13 splitString()

```
std::vector< std::string > cm::splitString (
            std::string string,
            pl::string_view splitBy )
```

Splits `string` by `splitBy`.

Parameters

string	The string to split.
splitBy	What to split `string` by.

Returns

The resulting strings.

Definition at line 8 of file split_string.cpp.

Here is the caller graph for this function:

5.3.2.14 toDataSetIdentifier()

```
DataSetIdentifier cm::toDataSetIdentifier (
            const cl::fs::Path & path )
```

Converts a path to a CSV file to the corresponding DataSetIdentifier.

Parameters

path	The path.

Returns

The resulting DataSetIdentifier.

Exceptions

cl::Exception	if `path` is unrecognized.

Definition at line 33 of file data_set_identifier.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

5.3.3 Variable Documentation

5.3.3.1 imuCount

```
constexpr std::size_t cm::imuCount  [inline], [constexpr]
```

Initial value:

```
{0
#define CM_IMU_X(enm)
                                        CM_IMU
}
```

Definition at line 20 of file imu.hpp.

5.3.3.2 imus

```
constexpr std::array<Imu, imuCount> cm::imus  [inline], [constexpr]
```

Initial value:

```
{{
#define CM_IMU_X(enm)
  CM_IMU
}}
```

Definition at line 26 of file imu.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, "Lukas_liegestuetzen1", 24)
- CS_SPECIALIZE_DATA_SET_INFO (Lucas2, "Lukas_liegestuetzen2", 19)
- CS_SPECIALIZE_DATA_SET_INFO (Lucas3, "Lukas_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 ,
            "Lukas_liegestuetzen1" ,
            24  )
```

5.4.2.13 CS_SPECIALIZE_DATA_SET_INFO() [13/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
            Lucas2 ,
            "Lukas_liegestuetzen2" ,
            19  )
```

5.4.2.14 CS_SPECIALIZE_DATA_SET_INFO() [14/20]

```
cs::CS_SPECIALIZE_DATA_SET_INFO (
            Lucas3 ,
            "Lukas_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

directoryPath	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

lhs	The left hand side operand.
rhs	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

os	The ostream to print to.
logInfo	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

os	The ostream to print to.
filterKind	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

os	The ostream to print to.
segmentationKind	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

lhs	The left hand side operand.
rhs	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

dataSet	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 accelerometer↩ Threshold, 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, ComparisonValue↩ Calculator comparisonValueCalculator)
- constexpr long double percentageOf (std::size_t amount, std::size_t totalCount) noexcept
- void runAboveThreshold (std::ostream &aboveThresholdLogOutputFileStream, 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 & aboveThresholdLogOutputFileStream,
            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 cm::Configuration::Builder Class Reference

Builder type for `Configuration`.

```
#include <configuration.hpp>
```

Public Member Functions

- Builder () noexcept

  Creates an empty `Builder`.
- Builder & skipWindow (bool value)

  Sets the skipWindow property.
- Builder & deleteTooClose (bool value)

  Sets the deleteTooClose property.
- Builder & deleteTooLowVariance (bool value)

  Sets the deleteTooLowVariance property.
- Builder & imu (Imu value)

  Sets the imu property.
- Builder & segmentationKind (std::string value)

  Sets the segmentationKind property.
- Builder & windowSize (std::size_t value)

  Sets the windowSize property.
- Builder & filterKind (std::string value)

  Sets the filterKind property.
- Configuration build () const

  Builds a `Configuration`.

6.1.1 Detailed Description

Builder type for `Configuration`.

Definition at line 38 of file configuration.hpp.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 Builder()

`cm::Configuration::Builder::Builder ( ) [noexcept]`

Creates an empty `Builder`.

Definition at line 36 of file configuration.cpp.

6.1.3 Member Function Documentation

6.1.3.1 build()

`Configuration cm::Configuration::Builder::build ( ) const`

Builds a `Configuration`.

Returns

The `Configuration` built.

Exceptions

cl::Exception	if one of the properties has not been set or is invalid.

Definition at line 90 of file configuration.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

6.1.3.2 deleteTooClose()

```
Configuration::Builder & cm::Configuration::Builder::deleteTooClose (
            bool value )
```

Sets the deleteTooClose property.

Parameters

value	The value to use.

Returns

*this

Definition at line 53 of file configuration.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

6.1.3.3 deleteTooLowVariance()

```
Configuration::Builder & cm::Configuration::Builder::deleteTooLowVariance (
            bool value )
```

Sets the deleteTooLowVariance property.

Parameters

value	The value to use.

Returns

*this

Definition at line 59 of file configuration.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

6.1.3.4 filterKind()

```
Configuration::Builder & cm::Configuration::Builder::filterKind (
            std::string value )
```

Sets the filterKind property.

Parameters

value	The value to use.

Returns

*this

Definition at line 84 of file configuration.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

6.1.3.5 imu()

```
Configuration::Builder & cm::Configuration::Builder::imu (
            Imu value )
```

Sets the imu property.

Parameters

value	The value to use.

Returns

*this

Definition at line 65 of file configuration.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

6.1.3.6 segmentationKind()

```
Configuration::Builder & cm::Configuration::Builder::segmentationKind (
            std::string value )
```

Sets the segmentationKind property.

Parameters

value	The value to use.

Returns

*this

Definition at line 71 of file configuration.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

6.1.3.7 skipWindow()

```
Configuration::Builder & cm::Configuration::Builder::skipWindow (
            bool value )
```

Sets the skipWindow property.

Parameters

value	The value to use.

Returns

*this

Definition at line 47 of file configuration.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

6.1.3.8 windowSize()

```
Configuration::Builder & cm::Configuration::Builder::windowSize (
            std::size_t value )
```

Sets the windowSize property.

Parameters

value	The value to use.

Returns

*this

Definition at line 78 of file configuration.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:

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

6.2 cl::col_traits< Col > Struct Template Reference

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

6.2.1 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

6.3 cm::Configuration Class Reference

Represents a possible configuration for the Python segmentor.

```
#include <configuration.hpp>
```

Classes

- class Builder

  Builder type for `Configuration`.

Public Member Functions

- bool skipWindow () const noexcept

  Read accessor for the skipWindow property.
- bool deleteTooClose () const noexcept

  Read accessor for the deleteTooClose property.
- bool deleteTooLowVariance () const noexcept

  Read accessor for the deleteTooLowVariance property.
- Imu imu () const noexcept

  Read accessor for the imu property.
- const std::string & segmentationKind () const noexcept

  Read accessor for the segmentationKind property.
- std::size_t windowSize () const noexcept

  Read accessor for the windowSize property.
- const std::string & filterKind () const noexcept

  Read accessor for the filterKind property.

Static Public Member Functions

- static const std::deque< bool > & skipWindowOptions () noexcept

  Returns the possible skipWindow options.
- static const std::deque< bool > & deleteTooCloseOptions () noexcept

  Returns the possible deleteTooClose options.
- static const std::deque< bool > & deleteTooLowVarianceOptions () noexcept

  Returns the possible deleteTooLowVariance options.
- static const std::vector< Imu > & imuOptions () noexcept

  Returns the possible imu options.
- static const std::vector< std::string > & segmentationKindOptions () noexcept

  Returns the possible segmentationKind options.
- static const std::vector< std::size_t > & windowSizeOptions () noexcept

  Returns the possible windowSize options.
- static const std::vector< std::string > & filterKindOptions () noexcept

  Returns the possible filterKind options.

Friends

- class Builder
- struct std::hash< Configuration >
- bool operator== (const Configuration &lhs, const Configuration &rhs) noexcept

  Compares two Configurations for equality.
- bool operator!= (const Configuration &lhs, const Configuration &rhs) noexcept

  Compares two Configurations for inequality.
- std::ostream & operator<< (std::ostream &os, const Configuration &config)

  Prints `config` to `os`.

6.3.1 Detailed Description

Represents a possible configuration for the Python segmentor.

Definition at line 30 of file configuration.hpp.

6.3.2 Member Function Documentation

6.3.2.1 deleteTooClose()

```
bool cm::Configuration::deleteTooClose ( ) const  [noexcept]
```

Read accessor for the deleteTooClose property.

Returns

The deleteTooClose option.

Definition at line 234 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.2 deleteTooCloseOptions()

```
const std::deque< bool > & cm::Configuration::deleteTooCloseOptions ( ) [static], [noexcept]
```

Returns the possible deleteTooClose options.

Returns

The deleteTooClose options.

Definition at line 148 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.3 deleteTooLowVariance()

```
bool cm::Configuration::deleteTooLowVariance ( ) const [noexcept]
```

Read accessor for the deleteTooLowVariance property.

Returns

The deleteTooLowVariance option.

Definition at line 236 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.4 deleteTooLowVarianceOptions()

```
const std::deque< bool > & cm::Configuration::deleteTooLowVarianceOptions ( ) [static], [noexcept]
```

Returns the possible deleteTooLowVariance options.

Returns

The deleteTooLowVariance options.

Definition at line 154 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.5 filterKind()

```
const std::string & cm::Configuration::filterKind ( ) const [noexcept]
```

Read accessor for the filterKind property.

Returns

The filterKind option.

Definition at line 250 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.6 filterKindOptions()

```
const std::vector< std::string > & cm::Configuration::filterKindOptions ( ) [static], [noexcept]
```

Returns the possible filterKind options.

Returns

The filterKind options.

Definition at line 180 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.7 imu()

```
Imu cm::Configuration::imu ( ) const [noexcept]
```

Read accessor for the imu property.

Returns

The imu option.

Definition at line 241 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.8 imuOptions()

```
const std::vector< Imu > & cm::Configuration::imuOptions ( ) [static], [noexcept]
```

Returns the possible imu options.

Returns

The imu options.

Definition at line 160 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.9 segmentationKind()

```
const std::string & cm::Configuration::segmentationKind ( ) const [noexcept]
```

Read accessor for the segmentationKind property.

Returns

The segmentationKind option.

Definition at line 243 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.10 segmentationKindOptions()

```
const std::vector< std::string > & cm::Configuration::segmentationKindOptions ( ) [static], [noexcept]
```

Returns the possible segmentationKind options.

Returns

The segmentationKind options.

Definition at line 167 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.11 skipWindow()

```
bool cm::Configuration::skipWindow ( ) const [noexcept]
```

Read accessor for the skipWindow property.

Returns

The skipWindow option.

Definition at line 232 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.12 skipWindowOptions()

```
const std::deque< bool > & cm::Configuration::skipWindowOptions ( ) [static], [noexcept]
```

Returns the possible skipWindow options.

Returns

The skipWindow options.

Definition at line 142 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.13 windowSize()

```
std::size_t cm::Configuration::windowSize ( ) const [noexcept]
```

Read accessor for the windowSize property.

Returns

The windowSize option.

Definition at line 248 of file configuration.cpp.

Here is the caller graph for this function:

6.3.2.14 windowSizeOptions()

```
const std::vector< std::size_t > & cm::Configuration::windowSizeOptions ( ) [static], [noexcept]
```

Returns the possible windowSize options.

Returns

The windowSize options.

Definition at line 173 of file configuration.cpp.

Here is the caller graph for this function:

6.3.3 Friends And Related Function Documentation

6.3.3.1 Builder

```
friend class Builder [friend]
```

Definition at line 32 of file configuration.hpp.

6.3.3.2 operator"!=

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

Compares two Configurations for inequality.

Parameters

lhs	The first operand.
rhs	The second operand.

Returns

true if `lhs` and `rhs` are considered not to be equal.

Definition at line 206 of file configuration.cpp.

6.3.3.3 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            const Configuration & config )  [friend]
```

Prints `config` to `os`.

Parameters

os	The ostream to print to.
config	The `Configuration` to print.

Returns

`os`

Definition at line 211 of file configuration.cpp.

6.3.3.4 operator==

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

Compares two Configurations for equality.

Parameters

lhs	The first operand.
rhs	The second operand.

Returns

true if `lhs` and `rhs` are considered to be equal.

Definition at line 186 of file configuration.cpp.

6.3.3.5 std::hash< Configuration >

```
friend struct std::hash< Configuration >  [friend]
```

Definition at line 33 of file configuration.hpp.

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

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

6.4 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.

6.4.1 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.

6.4.2 Member Typedef Documentation

6.4.2.1 this_type

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

Definition at line 23 of file csv_line.hpp.

6.4.3 Constructor & Destructor Documentation

6.4.3.1 CsvLineBuilder()

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

Creates an empty, invalid CsvLineBuilder.

Definition at line 44 of file csv_line.cpp.

6.4.4 Member Function Documentation

6.4.4.1 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:

6.4.4.2 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:

6.4.4.3 deleteLowVariance()

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

Write accessor for the delete low variance property.

Parameters

value	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:

6.4.4.4 deleteTooClose()

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

Write accessor for the delete too close property.

Parameters

value	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:

6.4.4.5 filter()

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

Write accessor for the filter property.

Parameters

value	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:

6.4.4.6 isOld()

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

Write accessor for the is old property.

Parameters

value	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:

6.4.4.7 kind()

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

Write accessor for the kind property.

Parameters

value	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:

6.4.4.8 repetitions()

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

Write accessor for the repetitions property.

Parameters

value	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:

6.4.4.9 segmentationPoints()

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

Write accessor for the segmentation points property.

Parameters

value	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:

6.4.4.10 sensor()

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

Write accessor for the sensor property.

Parameters

value	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:

6.4.4.11 skipWindow()

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

Write accessor for the skip window property.

Parameters

value	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:

6.4.4.12 windowSize()

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

Write accessor for the window size property.

Parameters

value	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

6.5 cl::data_set_accessor< Chan > Struct Template Reference

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

6.5.1 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

6.6 cs::data_set_info< Tag > Struct Template Reference

Meta function for data set tags.

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

6.6.1 Detailed Description

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

Meta function for data set tags.

Template Parameters

Tag	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

6.7 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)

6.7.1 Detailed Description

Definition at line 10 of file data_point.hpp.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 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:

6.7.3 Member Function Documentation

6.7.3.1 channel()

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

Definition at line 41 of file data_point.cpp.

Here is the caller graph for this function:

6.7.3.2 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:

6.7.3.3 sensor()

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

Definition at line 39 of file data_point.cpp.

Here is the caller graph for this function:

6.7.3.4 time()

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

Definition at line 37 of file data_point.cpp.

Here is the caller graph for this function:

6.7.3.5 value()

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

Definition at line 43 of file data_point.cpp.

Here is the caller graph for this function:

6.7.4 Friends And Related Function Documentation

6.7.4.1 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

6.8 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)

6.8.1 Detailed Description

Definition at line 14 of file data_set.hpp.

6.8.2 Member Typedef Documentation

6.8.2.1 ChannelAccessor

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

Definition at line 17 of file data_set.hpp.

6.8.2.2 size_type

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

Definition at line 16 of file data_set.hpp.

6.8.3 Member Function Documentation

6.8.3.1 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:

6.8.3.2 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:

6.8.3.3 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:

6.8.3.4 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:

6.8.3.5 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:

6.8.3.6 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:

6.8.3.7 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:

6.8.3.8 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:

6.8.3.9 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:

6.8.3.10 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:

6.8.3.11 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:

6.8.3.12 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:

6.8.3.13 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:

6.8.3.14 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:

6.8.3.15 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:

6.8.3.16 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:

6.8.3.17 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

6.9 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)

6.9.1 Detailed Description

Definition at line 23 of file error.hpp.

6.9.2 Member Enumeration Documentation

6.9.2.1 Kind

```
enum cl::Error::Kind
```

Enumerator

CL_ERROR_KIND	

Definition at line 26 of file error.hpp.

6.9.3 Constructor & Destructor Documentation

6.9.3.1 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.

6.9.4 Member Function Documentation

6.9.4.1 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:

6.9.4.2 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:

6.9.4.3 kind()

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

Definition at line 55 of file error.cpp.

Here is the caller graph for this function:

6.9.4.4 line()

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

Definition at line 61 of file error.cpp.

6.9.4.5 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:

6.9.4.6 raise()

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

Definition at line 65 of file error.cpp.

6.9.4.7 to_string()

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

Definition at line 74 of file error.cpp.

6.9.5 Friends And Related Function Documentation

6.9.5.1 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

6.10 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

6.10.1 Detailed Description

Definition at line 14 of file exception.hpp.

6.10.2 Member Typedef Documentation

6.10.2.1 base_type

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

Definition at line 16 of file exception.hpp.

6.10.3 Constructor & Destructor Documentation

6.10.3.1 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.

6.10.3.2 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.

6.10.4 Member Function Documentation

6.10.4.1 file()

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

Definition at line 30 of file exception.cpp.

Here is the caller graph for this function:

6.10.4.2 function()

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

Definition at line 32 of file exception.cpp.

Here is the caller graph for this function:

6.10.4.3 line()

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

Definition at line 34 of file exception.cpp.

Here is the caller graph for this function:

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

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

6.11 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 &copyToPath) 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.

6.11.1 Detailed Description

Represents a file.

Definition at line 11 of file file.hpp.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 File()

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

Creates a File from the given `path`.

Parameters

path	The path to use.

Definition at line 21 of file file.cpp.

Here is the call graph for this function:

6.11.3 Member Function Documentation

6.11.3.1 copyTo()

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

Copies this file in the filesystem.

Parameters

copyToPath	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:

6.11.3.2 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:

6.11.3.3 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:

6.11.3.4 moveTo()

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

Moves this file in the filesystem.

Parameters

newPath	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:

6.11.3.5 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:

6.11.3.6 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:

6.11.3.7 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

6.12 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.

6.12.1 Detailed Description

A binary file stream.

Definition at line 19 of file file_stream.hpp.

6.12.2 Member Typedef Documentation

6.12.2.1 this_type

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

Definition at line 30 of file file_stream.hpp.

6.12.3 Member Enumeration Documentation

6.12.3.1 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.

6.12.4 Constructor & Destructor Documentation

6.12.4.1 FileStream()

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

Move constructs from `other`.

Parameters

other	The file stream to move construct from.

Definition at line 70 of file file_stream.cpp.

6.12.4.2 ~FileStream()

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

Closes this file stream.

Definition at line 84 of file file_stream.cpp.

6.12.5 Member Function Documentation

6.12.5.1 create()

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

Creates a file stream.

Parameters

file	The file to open.
openMode	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:

6.12.5.2 operator=()

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

Move assigns `other` to this file stream.

Parameters

other	The file stream to move assign to this file stream.

Returns

*this

Definition at line 77 of file file_stream.cpp.

6.12.5.3 PL_NONCOPYABLE()

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

6.12.5.4 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.

6.12.5.5 write()

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

Writes data to the file.

Parameters

data	Pointer to the beginning of the memory region to write.
byteCount	The amount of bytes to write, starting from `data`.

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

6.13 std::hash<::cl::fs::Path > Struct Reference

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

Public Member Functions

- size_t operator() (const ::cl::fs::Path &path) const

6.13.1 Detailed Description

Definition at line 85 of file path.hpp.

6.13.2 Member Function Documentation

6.13.2.1 operator()()

```
size_t std::hash<::cl::fs::Path >::operator() (
            const ::cl::fs::Path & path ) const  [inline]
```

Definition at line 86 of file path.hpp.

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

- csv_lib/include/cl/fs/path.hpp

6.14 std::hash<::cm::Configuration > Struct Reference

```
#include <configuration.hpp>
```

Public Member Functions

- size_t operator() (const ::cm::Configuration &configuration) const

6.14.1 Detailed Description

Definition at line 248 of file configuration.hpp.

6.14.2 Member Function Documentation

6.14.2.1 operator()()

```
size_t std::hash<::cm::Configuration >::operator() (
            const ::cm::Configuration & configuration ) const  [inline]
```

Definition at line 249 of file configuration.hpp.

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

- confusion_matrix/include/configuration.hpp

6.15 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(0xFFFFFFFFFFFFFFFF)

  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.

6.15.1 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.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 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.

6.15.3 Member Function Documentation

6.15.3.1 create()

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

Creates a LogInfo from the given log file path.

Parameters

logFilePath	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:

6.15.3.2 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.

6.15.3.3 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.

6.15.3.4 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.

6.15.3.5 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.

6.15.3.6 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:

6.15.3.7 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.

6.15.3.8 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.

6.15.3.9 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.

6.15.3.10 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.

6.15.4 Friends And Related Function Documentation

6.15.4.1 operator"!=

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

Compares two LogInfos for inequality.

Parameters

lhs	The left hand side operand.
rhs	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.

6.15.4.2 operator<<

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

Prints a LogInfo to an ostream.

Parameters

os	The ostream to print to.
logInfo	The LogInfo to print.

Returns

os

Definition at line 292 of file log_info.cpp.

6.15.4.3 operator==

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

Compares two LogInfos for equality.

Parameters

lhs	The left hand side operand.
rhs	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.

6.15.5 Member Data Documentation

6.15.5.1 invalidSensor

```
const std::uint64_t cs::LogInfo::invalidSensor = UINT64_C(0xFFFFFFFFFFFFFFFF)  [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

6.16 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 acccessor 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(0xFFFFFFFFFFFFFFFF)

  Indicates an invalid sensor.

6.16.1 Detailed Description

A line out of a log file.

Definition at line 14 of file log_line.hpp.

6.16.2 Member Function Documentation

6.16.2.1 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:

6.16.2.2 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:

6.16.2.3 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

line	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:

6.16.2.4 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.

6.16.2.5 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.

6.16.3 Member Data Documentation

6.16.3.1 invalidSensor

```
const std::uint64_t cs::LogLine::invalidSensor = UINT64_C(0xFFFFFFFFFFFFFFFF)  [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

6.17 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.

Friends

- bool operator== (const ManualSegmentationPoint &lhs, const ManualSegmentationPoint &rhs) noexcept

  Compares two manual segmentation points for equality.
- bool operator!= (const ManualSegmentationPoint &lhs, const ManualSegmentationPoint &rhs) noexcept

  Compares two manual segmentation points for inequality.
- std::ostream & operator<< (std::ostream &os, const ManualSegmentationPoint &manualSegmentationPoint)

  Prints `manualSegmentationPoint` *to* `os`.

6.17.1 Detailed Description

Type used to represent a manual segmentation point.

Definition at line 26 of file manual_segmentation_point.hpp.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 ManualSegmentationPoint()

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

Creates a ManualSegmentationPoint.

Parameters

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

Exceptions

cl::Exception	if one of the arguments is out of bounds.

Definition at line 303 of file manual_segmentation_point.cpp.

Here is the call graph for this function:

6.17.3 Member Function Documentation

6.17.3.1 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 361 of file manual_segmentation_point.cpp.

6.17.3.2 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 356 of file manual_segmentation_point.cpp.

Here is the caller graph for this function:

6.17.3.3 hour()

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

Read accessor for the hour property.

Returns

The hour.

Definition at line 344 of file manual_segmentation_point.cpp.

Here is the caller graph for this function:

6.17.3.4 minute()

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

Read accessor for the minute property.

Returns

The minute.

Definition at line 346 of file manual_segmentation_point.cpp.

Here is the caller graph for this function:

6.17.3.5 readCsvFile()

```
std::unordered_map< DataSetIdentifier, std::vector< ManualSegmentationPoint > > cm::Manual↩
SegmentationPoint::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

cl::Exception	if parsing fails, CSV processing fails or the CSV file is missing.

Definition at line 161 of file manual_segmentation_point.cpp.

Here is the caller graph for this function:

6.17.3.6 second()

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

Read accessor for the second property.

Returns

The second.

Definition at line 351 of file manual_segmentation_point.cpp.

Here is the caller graph for this function:

6.17.4 Friends And Related Function Documentation

6.17.4.1 operator"!=

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

Compares two manual segmentation points for inequality.

Parameters

lhs	The first operand.
rhs	The second operand.

Returns

true if `lhs` is considered not equal to `rhs`; false otherwise.

Definition at line 139 of file manual_segmentation_point.cpp.

6.17.4.2 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            const ManualSegmentationPoint & manualSegmentationPoint )  [friend]
```

Prints `manualSegmentationPoint` to `os`.

Parameters

os	The ostream to print to
manualSegmentationPoint	The `ManualSegmentationPoint` to print.

Returns

os

Definition at line 146 of file manual_segmentation_point.cpp.

6.17.4.3 operator==

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

Compares two manual segmentation points for equality.

Parameters

lhs	The first operand.
rhs	The second operand.

Returns

true if `lhs` is considered equal to `rhs`; false otherwise.

Definition at line 131 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

6.18 cl::fs::Path Class Reference

A filesystem path.

`#include <path.hpp>`

Public Member Functions

- PL_IMPLICIT Path (std::string path)

  Creates a path.
- PL_IMPLICIT Path (const char *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`.

6.18.1 Detailed Description

A filesystem path.

Definition at line 14 of file path.hpp.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 Path() [1/2]

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

Creates a path.

Parameters

path	The string to construct from.

Definition at line 37 of file path.cpp.

6.18.2.2 Path() [2/2]

```
cl::fs::Path::Path (
            const char * path )
```

Creates a path.

Parameters

path	The string to construct from.

Definition at line 44 of file path.cpp.

6.18.3 Member Function Documentation

6.18.3.1 exists()

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

Checks if the path exists.

Returns

true if the path exists; otherwise false.

Definition at line 46 of file path.cpp.

Here is the call graph for this function:

6.18.3.2 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 102 of file path.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

6.18.3.3 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 75 of file path.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

6.18.3.4 str()

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

Read accessor for the underlying string.

Returns

The underlying string.

Definition at line 125 of file path.cpp.

Here is the caller graph for this function:

6.18.4 Friends And Related Function Documentation

6.18.4.1 operator<

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

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

Parameters

lhs	The left hand side operand.
rhs	The right hand side operand.

Returns

true if lhs < rhs; otherwise false.

Definition at line 27 of file path.cpp.

6.18.4.2 operator<<

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

Prints a Path to an ostream.

Parameters

os	the ostream to print to.
path	The path to print.

Returns

`os`

Definition at line 22 of file path.cpp.

6.18.4.3 operator==

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

Equality compares `lhs` and `rhs`.

Parameters

lhs	The left hand side operand.
rhs	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

6.19 cl::Process Class Reference

```
#include <process.hpp>
```

Public Types

- using this_type = Process

Public Member Functions

- PL_NONCOPYABLE (Process)
- Process (this_type &&other) noexcept
- this_type & operator= (this_type &&other) noexcept
- ~Process ()
- std::FILE * file () noexcept
- const std::FILE * file () const noexcept

Static Public Member Functions

- static Expected< Process > create (pl::string_view command, pl::string_view mode)

6.19.1 Detailed Description

Definition at line 11 of file process.hpp.

6.19.2 Member Typedef Documentation

6.19.2.1 this_type

```
using cl::Process::this_type = Process
```

Definition at line 15 of file process.hpp.

6.19.3 Constructor & Destructor Documentation

6.19.3.1 Process()

```
cl::Process::Process (
            this_type && other )  [noexcept]
```

Definition at line 56 of file process.cpp.

6.19.3.2 ~Process()

```
cl::Process::~Process ( )
```

Definition at line 69 of file process.cpp.

6.19.4 Member Function Documentation

6.19.4.1 create()

```
Expected< Process > cl::Process::create (
            pl::string_view command,
            pl::string_view mode )  [static]
```

Definition at line 36 of file process.cpp.

Here is the caller graph for this function:

6.19.4.2 file() [1/2]

```
const std::FILE* cl::Process::file ( ) const  [noexcept]
```

6.19.4.3 file() [2/2]

```
const std::FILE * cl::Process::file ( )  [noexcept]
```

Definition at line 79 of file process.cpp.

6.19.4.4 operator=()

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

Definition at line 61 of file process.cpp.

6.19.4.5 PL_NONCOPYABLE()

```
cl::Process::PL_NONCOPYABLE (
            Process  )
```

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

- csv_lib/include/cl/process.hpp
- csv_lib/src/cl/process.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/process.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.cpp 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/process.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/configuration.hpp include/create_segmentation_results.hpp include/data_set_identifier.hpp include/imu.hpp include/interpolated_data_set_paths.hpp include/manual_segmentation_point.hpp include/python_output.hpp include/segment.hpp include/split_string.hpp) set(LIB_SOURCES src/configuration.cpp src/create_segmentation_results.cpp src/data_set_identifier.cpp src/imu.cpp src/interpolated_data_set_paths.cpp src/manual_segmentation_point.cpp src/python_output.cpp src/segment.cpp src/split_string.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 data_set_identifier_test.cpp interpolated_data_set_paths_test.cpp main.cpp manual_segmentation_point_test.cpp segment_test.cpp split_string_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, "Lukas_liegestuetzen1", 24)
- cs::CS_SPECIALIZE_DATA_SET_INFO (Lucas2, "Lukas_liegestuetzen2", 19)
- cs::CS_SPECIALIZE_DATA_SET_INFO (Lucas3, "Lukas_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<>                                                                    \
  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 <iosfwd>
```

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 <cl/error.hpp>
#include <cl/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 <iosfwd>
#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 <iosfwd>
#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 <ostream>
#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 <ostream>
#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:

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 <cstdlib>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/unused.hpp>
#include "create_segmentation_results.hpp"
#include "manual_segmentation_point.hpp"
```

Include dependency graph for main.cpp:

Functions

- int main (int argc, char ∗argv[ ])

7.32.1 Function Documentation

7.32.1.1 main()

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

Definition at line 11 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 <ostream>
#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, shouldWorkWithOldPath)
- TEST (LogInfo, shouldWorkWithOldPath2)
- TEST (LogInfo, shouldResultInErrorIfLogFilePathIsTooShort)

- TEST (LogInfo, shouldFailIfSkipWindowIsInvalid)
- TEST (LogInfo, shouldFailIfDeleteTooCloseIsInvalid)
- TEST (LogInfo, shouldFailIfDeleteTooLowVarianceIsInvalid)
- TEST (LogInfo, shouldFailIfSegmentationKindIsInvalid)
- TEST (LogInfo, shouldFailIfWindowSizeIsInvalid)
- TEST (LogInfo, shouldFailIfFilterIsInvalid)
- TEST (LogInfo, shouldCreateUninitialzedObjectWhenDefaultConstructorIsCalled)

7.38.1 Function Documentation

7.38.1.1 TEST() [1/19]

```
TEST (
            LogInfo ,
            shouldCreateUninitialzedObjectWhenDefaultConstructorIsCalled  )
```

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 ,
            shouldFailIfSkipWindowIsInvalid  )
```

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 ,
            shouldFailIfWindowSizeIsInvalid  )
```

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/configuration.hpp File Reference

```
#include <cstddef>
#include <deque>
#include <iosfwd>
#include <string>
#include <vector>
#include <tl/optional.hpp>
#include <pl/hash.hpp>
```

```
#include "imu.hpp"
```

Include dependency graph for configuration.hpp:

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

Classes

- class cm::Configuration

  Represents a possible configuration for the Python segmentor.
- class cm::Configuration::Builder

  Builder type for `Configuration`.
- struct std::hash<::cm::Configuration >

Namespaces

- cm

7.41 confusion_matrix/include/create_segmentation_results.hpp File Reference

```
#include <cstdint>
#include <unordered_map>
#include <vector>
#include <cl/fs/path.hpp>
```

```
#include "configuration.hpp"
```

Include dependency graph for create_segmentation_results.hpp:

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

Namespaces

- cm

Functions

- std::unordered_map< cm::Configuration, std::unordered_map< cl::fs::Path, std::vector< std::uint64_t > > > cm::createSegmentationResults ()

7.42 confusion_matrix/include/data_set_identifier.hpp File Reference

```
#include <iosfwd>
#include <cl/fs/path.hpp>
```

Include dependency graph for data_set_identifier.hpp:

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

Namespaces

- cm

Macros

- #define CM_DATA_SET_IDENTIFIER
- #define CM_DATA_SET_IDENTIFIER_X(enm) enm,

Enumerations

- enum cm::DataSetIdentifier { cm::DataSetIdentifier::CM_DATA_SET_IDENTIFIER_X, cm::DataSetIdentifier::CM_DATA_SET_I
  }

Functions

- std::ostream & cm::operator<< (std::ostream &os, DataSetIdentifier dsi)

  Prints a DataSetIdentifier to an ostream.
- DataSetIdentifier cm::toDataSetIdentifier (const cl::fs::Path &path)

  Converts a path to a CSV file to the corresponding DataSetIdentifier.

7.42.1 Macro Definition Documentation

7.42.1.1 CM_DATA_SET_IDENTIFIER

```
#define CM_DATA_SET_IDENTIFIER
```

Value:

```
CM_DATA_SET_IDENTIFIER_X(Felix_11_17_39) \
CM_DATA_SET_IDENTIFIER_X(Felix_12_50_00) \
CM_DATA_SET_IDENTIFIER_X(Felix_13_00_09) \
CM_DATA_SET_IDENTIFIER_X(Mike_14_07_33)  \
CM_DATA_SET_IDENTIFIER_X(Mike_14_14_32)  \
CM_DATA_SET_IDENTIFIER_X(Mike_14_20_28)  \
CM_DATA_SET_IDENTIFIER_X(Marsi_14_59_59) \
CM_DATA_SET_IDENTIFIER_X(Marsi_15_13_22) \
CM_DATA_SET_IDENTIFIER_X(Marsi_15_31_36) \
CM_DATA_SET_IDENTIFIER_X(Jan_1)          \
CM_DATA_SET_IDENTIFIER_X(Jan_2)          \
CM_DATA_SET_IDENTIFIER_X(Jan_3)          \
CM_DATA_SET_IDENTIFIER_X(Andre_1)        \
CM_DATA_SET_IDENTIFIER_X(Andre_2)        \
CM_DATA_SET_IDENTIFIER_X(Andre_3)        \
CM_DATA_SET_IDENTIFIER_X(Andre_Squats_1) \
CM_DATA_SET_IDENTIFIER_X(Andre_Squats_2) \
CM_DATA_SET_IDENTIFIER_X(Lucas_1)        \
CM_DATA_SET_IDENTIFIER_X(Lucas_2)        \
CM_DATA_SET_IDENTIFIER_X(Lucas_3)
```

Definition at line 8 of file data_set_identifier.hpp.

7.42.1.2 CM_DATA_SET_IDENTIFIER_X

```
#define CM_DATA_SET_IDENTIFIER_X(
              enm ) enm,
```

Definition at line 31 of file data_set_identifier.hpp.

7.43 confusion_matrix/include/imu.hpp File Reference

```
#include <cstddef>
#include <array>
#include <iosfwd>
```

Include dependency graph for imu.hpp:

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

Namespaces

- cm

Macros

- #define CM_IMU
- #define CM_IMU_X(enm) enm,
- #define CM_IMU_X(enm) +1
- #define CM_IMU_X(enm) ::cm::Imu::enm,

Enumerations

- enum cm::Imu { cm::Imu::CM_IMU_X, cm::Imu::CM_IMU }

Functions

- std::ostream & cm::operator<< (std::ostream &os, Imu imu)

Variables

- constexpr std::size_t cm::imuCount
- constexpr std::array< Imu, imuCount > cm::imus

7.43.1 Macro Definition Documentation

7.43.1.1 CM_IMU

```
#define CM_IMU
```

Value:

```
CM_IMU_X(Accelerometer) \
CM_IMU_X(Gyroscope)
```

Definition at line 10 of file imu.hpp.

7.43.1.2 CM_IMU_X [1/3]

```
#define CM_IMU_X(
            enm ) enm,
```

Definition at line 15 of file imu.hpp.

7.43.1.3 CM_IMU_X [2/3]

```
#define CM_IMU_X(
            enm ) +1
```

Definition at line 15 of file imu.hpp.

7.43.1.4 CM_IMU_X [3/3]

```
#define CM_IMU_X(
            enm ) ::cm::Imu::enm,
```

Definition at line 15 of file imu.hpp.

7.44 confusion_matrix/include/interpolated_data_set_paths.hpp File Reference

```
#include <vector>
#include <cl/fs/path.hpp>
```

Include dependency graph for interpolated_data_set_paths.hpp:

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

Namespaces

- cm

Functions

- std::vector< cl::fs::Path > cm::interpolatedDataSetPaths ()

  Returns the paths to the interpolated data sets.

7.45 confusion_matrix/include/manual_segmentation_point.hpp File Reference

```
#include <cstdint>
#include <iosfwd>
#include <unordered_map>
#include <vector>
#include <cl/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

7.46 confusion_matrix/include/python_output.hpp File Reference

```
#include <cl/fs/path.hpp>
#include "configuration.hpp"
```

Include dependency graph for python_output.hpp:

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

Namespaces

- cm

Functions

- std::string cm::pythonOutput (const cl::fs::Path &csvFilePath, const Configuration &segmentorConfiguration)

  Runs the Python segmentor on `path`.

7.47 confusion_matrix/include/segment.hpp File Reference

```
#include <cstdint>
#include <unordered_map>
#include <vector>
#include <cl/fs/path.hpp>
```

```
#include "configuration.hpp"
```

Include dependency graph for segment.hpp:

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

Namespaces

- cm

Functions

- std::unordered_map< cl::fs::Path, std::vector< std::uint64_t > > cm::segment (const Configuration &segmentorConfiguration)

7.48 confusion_matrix/include/split_string.hpp File Reference

```
#include <string>
#include <vector>
```

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

Include dependency graph for split_string.hpp:

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

Namespaces

- cm

Functions

- std::vector< std::string > cm::splitString (std::string string, pl::string_view splitBy)

  Splits `string` *by* `splitBy`.

7.49 confusion_matrix/src/configuration.cpp File Reference

```
#include <algorithm>
#include <ostream>
#include <tuple>
#include <utility>
#include <fmt/format.h>
#include <fmt/ostream.h>
```

```
#include <pl/begin_end_macro.hpp>
#include <cl/exception.hpp>
#include "configuration.hpp"
```

Include dependency graph for configuration.cpp:

Namespaces

- cm

Macros

- #define CM_ENSURE_HAS_VALUE(dataMember)
- #define CM_CONTAINS(container, value) contains(container.begin(), container.end(), value)
- #define CM_ENSURE_CONTAINS(container, dataMember)

Functions

- bool cm::operator== (const Configuration &lhs, const Configuration &rhs) noexcept
- bool cm::operator!= (const Configuration &lhs, const Configuration &rhs) noexcept
- std::ostream & cm::operator<< (std::ostream &os, const Configuration &config)

7.49.1 Macro Definition Documentation

7.49.1.1 CM_CONTAINS

```
#define CM_CONTAINS(
            container,
            value ) contains(container.begin(), container.end(), value)
```

7.49.1.2 CM_ENSURE_CONTAINS

```
#define CM_ENSURE_CONTAINS(
            container,
            dataMember )
```

Value:

```
  PL_BEGIN_MACRO                                                          \
  if (!CM_CONTAINS(container, dataMember)) {                              \
    CL_THROW_FMT(                                                         \
      "\"{}\" is not a valid option for \"{}\"", *dataMember, #dataMember); \
  }                                                                       \
  PL_END_MACRO
```

7.49.1.3 CM_ENSURE_HAS_VALUE

```
#define CM_ENSURE_HAS_VALUE(
            dataMember )
```

Value:

```
PL_BEGIN_MACRO                                          \
if (!dataMember.has_value()) {                          \
  CL_THROW_FMT("\"{}\" was nullopt!", #dataMember);     \
}                                                       \
PL_END_MACRO
```

7.50 confusion_matrix/src/create_segmentation_results.cpp File Reference

```
#include "create_segmentation_results.hpp"
#include "segment.hpp"
```

Include dependency graph for create_segmentation_results.cpp:

Namespaces

- cm

Functions

- std::unordered_map< cm::Configuration, std::unordered_map< cl::fs::Path, std::vector< std::uint64_t > > > cm::createSegmentationResults ()

7.51 confusion_matrix/src/data_set_identifier.cpp File Reference

```
#include <ostream>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/string_view.hpp>
#include <pl/unreachable.hpp>
#include <cl/exception.hpp>
```

```
#include "data_set_identifier.hpp"
```

Include dependency graph for data_set_identifier.cpp:

Namespaces

- cm

Macros

- #define CM_DATA_SET_IDENTIFIER_X(enm) case DataSetIdentifier::enm: return #enm;
- #define DSI DataSetIdentifier

Functions

- std::ostream & cm::operator<< (std::ostream &os, DataSetIdentifier dsi)

  Prints a DataSetIdentifier to an ostream.
- DataSetIdentifier cm::toDataSetIdentifier (const cl::fs::Path &path)

  Converts a path to a CSV file to the corresponding DataSetIdentifier.

7.51.1 Macro Definition Documentation

7.51.1.1 CM_DATA_SET_IDENTIFIER_X

```
#define CM_DATA_SET_IDENTIFIER_X(
            enm ) case DataSetIdentifier::enm:  return #enm;
```

7.51.1.2 DSI

```
#define DSI DataSetIdentifier
```

7.52 confusion_matrix/src/imu.cpp File Reference

```
#include <ostream>
#include <pl/algo/ranged_algorithms.hpp>
#include <pl/unreachable.hpp>
#include "imu.hpp"
```

Include dependency graph for imu.cpp:

Namespaces

- cm

Macros

- #define CM_IMU_X(enm) case Imu::enm: return os << toLower(#enm);

Functions

- std::ostream & cm::operator<< (std::ostream &os, Imu imu)

7.52.1 Macro Definition Documentation

7.52.1.1 CM_IMU_X

```
#define CM_IMU_X(
            enm ) case Imu::enm:  return os << toLower(#enm);
```

7.53 confusion_matrix/src/interpolated_data_set_paths.cpp File Reference

```
#include <string>
#include <pl/algo/erase.hpp>
#include <pl/algo/ranged_algorithms.hpp>
#include <pl/string_view.hpp>
#include <cl/exception.hpp>
#include <cl/fs/directory_listing.hpp>
#include "interpolated_data_set_paths.hpp"
```

Include dependency graph for interpolated_data_set_paths.cpp:

Namespaces

- cm

Functions

- std::vector< cl::fs::Path > cm::interpolatedDataSetPaths ()

  Returns the paths to the interpolated data sets.

7.54 confusion_matrix/src/manual_segmentation_point.cpp File Reference

```
#include <cassert>
#include <cmath>
#include <ostream>
#include <fmt/format.h>
#include <fmt/ostream.h>
#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

Functions

- bool cm::operator== (const ManualSegmentationPoint &lhs, const ManualSegmentationPoint &rhs) noexcept
- bool cm::operator!= (const ManualSegmentationPoint &lhs, const ManualSegmentationPoint &rhs) noexcept
- std::ostream & cm::operator<< (std::ostream &os, const ManualSegmentationPoint &manual↩ SegmentationPoint)

7.54.1 Macro Definition Documentation

7.54.1.1 DSI

```
#define DSI DataSetIdentifier
```

7.55 confusion_matrix/src/python_output.cpp File Reference

```
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/os.hpp>
#include <pl/string_view.hpp>
#include <cl/process.hpp>
#include "python_output.hpp"
```

Include dependency graph for python_output.cpp:

Namespaces

- cm

Macros

- #define CM_SEGMENTOR "./preprocessed_segment.sh"
- #define CM_DEV_NULL "/dev/null"

Functions

- std::string cm::pythonOutput (const cl::fs::Path &csvFilePath, const Configuration &segmentorConfiguration)

  Runs the Python segmentor on `path`.

7.55.1 Macro Definition Documentation

7.55.1.1 CM_DEV_NULL

```
#define CM_DEV_NULL "/dev/null"
```

Definition at line 14 of file python_output.cpp.

7.55.1.2 CM_SEGMENTOR

```
#define CM_SEGMENTOR "./preprocessed_segment.sh"
```

Definition at line 13 of file python_output.cpp.

7.56 confusion_matrix/src/segment.cpp File Reference

```
#include <cl/exception.hpp>
#include <cl/s2n.hpp>
#include "interpolated_data_set_paths.hpp"
#include "python_output.hpp"
#include "segment.hpp"
#include "split_string.hpp"
```

Include dependency graph for segment.cpp:

Namespaces

- cm

Functions

- std::unordered_map< cl::fs::Path, std::vector< std::uint64_t > > cm::segment (const Configuration &segmentorConfiguration)

7.57 confusion_matrix/src/split_string.cpp File Reference

```
#include <cstring>
#include <mutex>
#include "split_string.hpp"
```

Include dependency graph for split_string.cpp:

Namespaces

- cm

Functions

- std::vector< std::string > cm::splitString (std::string string, pl::string_view splitBy)

  Splits `string` *by* `splitBy`.

7.58 confusion_matrix/test/data_set_identifier_test.cpp File Reference

```
#include <array>
#include "gtest/gtest.h"
#include <cl/fs/path.hpp>
```

```
#include "data_set_identifier.hpp"
```

Include dependency graph for data_set_identifier_test.cpp:

Macros

- #define DSI ::cm::DataSetIdentifier

Functions

- TEST (DataSetIdentifier, shouldConvertPaths)

7.58.1 Macro Definition Documentation

7.58.1.1 DSI

```
#define DSI ::cm::DataSetIdentifier
```

7.58.2 Function Documentation

7.58.2.1 TEST()

```
TEST (
            DataSetIdentifier ,
            shouldConvertPaths  )
```

Definition at line 9 of file data_set_identifier_test.cpp.

Here is the call graph for this function:

7.59 confusion_matrix/test/interpolated_data_set_paths_test.cpp File Reference

```
#include <string>
#include "gtest/gtest.h"
#include "interpolated_data_set_paths.hpp"
```

Include dependency graph for interpolated_data_set_paths_test.cpp:

Functions

- TEST (interpolatedDataSetPaths, shouldFetchPaths)

7.59.1 Function Documentation

7.59.1.1 TEST()

```
TEST (
            interpolatedDataSetPaths ,
            shouldFetchPaths  )
```

Definition at line 7 of file interpolated_data_set_paths_test.cpp.

Here is the call graph for this function:

7.60 confusion_matrix/test/manual_segmentation_point_test.cpp File Reference

```
#include <cstddef>
#include <array>
#include <ostream>
#include <sstream>
#include <string>
#include "gtest/gtest.h"
#include "manual_segmentation_point.hpp"
```

Include dependency graph for manual_segmentation_point_test.cpp:

Macros

- #define DSI ::cm::DataSetIdentifier

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)
- TEST (ManualSegmentationPoint, shouldBeAbleToImportCsvFile)
- TEST (ManualSegmentationPoint, shouldPrint)

7.60.1 Macro Definition Documentation

7.60.1.1 DSI

```
#define DSI ::cm::DataSetIdentifier
```

7.60.2 Function Documentation

7.60.2.1 TEST() [1/11]

```
TEST (
            ManualSegmentationPoint ,
            shouldBeAbleToImportCsvFile  )
```

Definition at line 98 of file manual_segmentation_point_test.cpp.

Here is the call graph for this function:

7.60.2.2 TEST() [2/11]

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

Definition at line 12 of file manual_segmentation_point_test.cpp.

7.60.2.3 TEST() [3/11]

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

Definition at line 82 of file manual_segmentation_point_test.cpp.

7.60.2.4 TEST() [4/11]

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

Definition at line 64 of file manual_segmentation_point_test.cpp.

7.60.2.5 TEST() [5/11]

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

Definition at line 70 of file manual_segmentation_point_test.cpp.

7.60.2.6 TEST() [6/11]

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

Definition at line 76 of file manual_segmentation_point_test.cpp.

7.60.2.7 TEST() [7/11]

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

Definition at line 58 of file manual_segmentation_point_test.cpp.

7.60.2.8 TEST() [8/11]

```
TEST (
            ManualSegmentationPoint ,
            shouldPrint  )
```

Definition at line 370 of file manual_segmentation_point_test.cpp.

7.60.2.9 TEST() [9/11]

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

Definition at line 46 of file manual_segmentation_point_test.cpp.

7.60.2.10 TEST() [10/11]

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

Definition at line 22 of file manual_segmentation_point_test.cpp.

7.60.2.11 TEST() [11/11]

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

Definition at line 34 of file manual_segmentation_point_test.cpp.

7.61 confusion_matrix/test/segment_test.cpp File Reference

```
#include <exception>
#include <stdexcept>
#include "gtest/gtest.h"
#include <pl/os.hpp>
#include "segment.hpp"
```

Include dependency graph for segment_test.cpp:

Macros

- #define EXPECT_SEGMENTATION_POINTS(path, ...) EXPECT_EQ((std::vector<std::uint64_t>{__VA_↩ ARGS__}), fetch(path))

Functions

- TEST (segment, shouldGetExpectedSegmentationPointsFromPython)

7.61.1 Macro Definition Documentation

7.61.1.1 EXPECT_SEGMENTATION_POINTS

```
#define EXPECT_SEGMENTATION_POINTS(
            path,
            ...  ) EXPECT_EQ((std::vector<std::uint64_t>{__VA_ARGS__}), fetch(path))
```

7.61.2 Function Documentation

7.61.2.1 TEST()

```
TEST (
            segment ,
            shouldGetExpectedSegmentationPointsFromPython  )
```

Definition at line 11 of file segment_test.cpp.

Here is the call graph for this function:

7.62 confusion_matrix/test/split_string_test.cpp File Reference

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

Include dependency graph for split_string_test.cpp:

Functions

- TEST (splitString, shouldSplitString)

7.62.1 Function Documentation

7.62.1.1 TEST()

```
TEST (
            splitString ,
            shouldSplitString  )
```

Definition at line 5 of file split_string_test.cpp.

Here is the call graph for this function:

7.63 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 accelerometer↩ Threshold, long double gyroscopeThreshold)

7.64 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)

7.65 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)

7.66 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, ComparisonValue↩ Calculator comparisonValueCalculator)

7.67 counting/include/percentage_of.hpp File Reference

```
#include <cstddef>
```

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

7.68 counting/include/run_above_threshold.hpp File Reference

```
#include <iosfwd>
#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 &aboveThresholdLogOutputFileStream, const cl::DataSet &dataSet)

7.69 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 accelerometer↩ Threshold, long double gyroscopeThreshold)

7.69.1 Macro Definition Documentation

7.69.1.1 CL_CHANNEL_X

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

7.69.2 Variable Documentation

7.69.2.1 channel

```
cl::Channel channel
```

Definition at line 18 of file above_threshold.cpp.

7.69.2.2 channelAccessor

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

Definition at line 17 of file above_threshold.cpp.

7.70 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)

7.71 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)

7.72 counting/src/run_above_threshold.cpp File Reference

```
#include <ostream>
#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 &aboveThresholdLogOutputFileStream, const cl::DataSet &dataSet)

7.73 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)

7.73.1 Macro Definition Documentation

7.73.1.1 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.

7.73.2 Function Documentation

7.73.2.1 TEST()

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

Definition at line 10 of file above_threshold_test.cpp.

Here is the call graph for this function:

7.74 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)

7.74.1 Macro Definition Documentation

7.74.1.1 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.

7.74.2 Function Documentation

7.74.2.1 TEST()

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

Definition at line 10 of file percentage_of_test.cpp.

Here is the call graph for this function:

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

```
#include <cstddef>
#include <cstdint>
#include <array>
#include <iosfwd>
#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_X, 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}

7.75.1 Macro Definition Documentation

7.75.1.1 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.

7.75.1.2 CL_CHANNEL_X [1/4]

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

Definition at line 41 of file channel.hpp.

7.75.1.3 CL_CHANNEL_X [2/4]

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

Definition at line 41 of file channel.hpp.

7.75.1.4 CL_CHANNEL_X [3/4]

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

Definition at line 41 of file channel.hpp.

7.75.1.5 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.

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

```
#include <cstddef>
#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

7.76.1 Macro Definition Documentation

7.76.1.1 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.

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

```
#include <iosfwd>
#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

7.78 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

7.79 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.

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

```
#include <cstddef>
#include <iosfwd>
#include <string>
#include <tl/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 = tl::expected< Ty, Error >

7.80.1 Macro Definition Documentation

7.80.1.1 CL_ERROR_KIND

```
#define CL_ERROR_KIND
```

Value:

```
CL_ERROR_KIND_X(Filesystem)       \
CL_ERROR_KIND_X(InvalidArgument) \
CL_ERROR_KIND_X(OutOfRange)       \
CL_ERROR_KIND_X(Parsing)          \
CL_ERROR_KIND_X(Logic)            \
CL_ERROR_KIND_X(OperatingSystem)
```

Definition at line 14 of file error.hpp.

7.80.1.2 CL_ERROR_KIND_X

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

Definition at line 27 of file error.hpp.

7.80.1.3 CL_UNEXPECTED

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

Value:

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

Definition at line 67 of file error.hpp.

7.81 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__))

7.81.1 Macro Definition Documentation

7.81.1.1 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.

7.81.1.2 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.

7.82 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.

7.83 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

7.84 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

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

```
#include <functional>
#include <iosfwd>
#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.
- struct std::hash<::cl::fs::Path >

Namespaces

- cl
- cl::fs

7.86 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.

7.86.1 Macro Definition Documentation

7.86.1.1 CL_FS_SEPARATOR

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

The filesystem separator of the operating system.

Definition at line 11 of file separator.hpp.

7.87 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-6 string.
- std::wstring cl::fs::formatError (DWORD errorCode)

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

7.87.1 Detailed Description

Contains Microsoft Windows specific functions.

7.88 csv_lib/include/cl/process.hpp File Reference

```
#include <cstdio>
#include <pl/noncopyable.hpp>
#include <pl/string_view.hpp>
#include "error.hpp"
```

Include dependency graph for process.hpp:

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

Classes

- class cl::Process

Namespaces

- cl

7.89 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

7.90 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)

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

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

```
#include <iosfwd>
```

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(enumerator, value) enumerator = value,
- #define CL_SENSOR_X(enm, v) ::cl::Sensor::enm,

Enumerations

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

Functions

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

Variables

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

7.91.1 Macro Definition Documentation

7.91.1.1 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.

7.91.1.2 CL_SENSOR_X [1/2]

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

Definition at line 16 of file sensor.hpp.

7.91.1.3 CL_SENSOR_X [2/2]

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

Definition at line 16 of file sensor.hpp.

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

```
#include <ios>
#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)

7.93 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 ()

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

```
#include <cassert>
#include <cinttypes>
#include <cstring>
#include <ostream>
#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)

7.94.1 Macro Definition Documentation

7.94.1.1 CL_CHANNEL_X [1/2]

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

7.94.1.2 CL_CHANNEL_X [2/2]

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

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

```
#include <ostream>
#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 dataPoint value ())

7.95.1 Function Documentation

7.95.1.1 channel()

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

7.95.1.2 fileName()

```
dataPoint fileName ( )
```

Here is the caller graph for this function:

7.95.1.3 sensor()

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

Here is the caller graph for this function:

7.95.1.4 time()

```
dataPoint dataPoint time ( )
```

7.95.1.5 value()

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

Here is the caller graph for this function:

7.96 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

7.97 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.

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

```
#include <cassert>
#include <ostream>
#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)

7.98.1 Macro Definition Documentation

7.98.1.1 CL_ERROR_KIND_X

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

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

```
#include <utility>
#include "cl/exception.hpp"
```

Include dependency graph for exception.cpp:

Namespaces

- cl

7.100 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.

7.101 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

7.102 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

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

```
#include <cassert>
#include <cstring>
#include <ostream>
#include <utility>
#include <pl/algo/ranged_algorithms.hpp>
#include <pl/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

7.104 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-6 string.
- std::wstring cl::fs::formatError (DWORD errorCode)

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

7.105 csv_lib/src/cl/process.cpp File Reference

```
#include <stdio.h>
#include <cassert>
#include <fmt/format.h>
#include <fmt/ostream.h>
#include <pl/algo/ranged_algorithms.hpp>
#include <pl/os.hpp>
```

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

Include dependency graph for process.cpp:

Namespaces

- cl

7.106 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

7.107 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)

7.107.1 Macro Definition Documentation

7.107.1.1 CL_SENSOR_X

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

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

```
#include <cstdio>
#include <ios>
#include <iostream>
#include "cl/use_unbuffered_io.hpp"
```

Include dependency graph for use_unbuffered_io.cpp:

Namespaces

- cl

Functions

- void cl::useUnbufferedIo ()

7.109 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)

7.109.1 Function Documentation

7.109.1.1 TEST() [1/4]

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

Definition at line 7 of file channel_test.cpp.

7.109.1.2 TEST() [2/4]

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

Definition at line 9 of file channel_test.cpp.

7.109.1.3 TEST() [3/4]

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

Definition at line 35 of file channel_test.cpp.

Here is the call graph for this function:

7.109.1.4 TEST() [4/4]

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

Definition at line 19 of file channel_test.cpp.

7.110 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)

7.110.1 Function Documentation

7.110.1.1 TEST() [1/2]

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

Definition at line 22 of file column_test.cpp.

7.110.1.2 TEST() [2/2]

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

Definition at line 7 of file column_test.cpp.

7.111 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

7.111.1 Function Documentation

7.111.1.1 TEST() [1/2]

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

Definition at line 23 of file data_point_test.cpp.

Here is the call graph for this function:

7.111.1.2 TEST() [2/2]

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

Definition at line 14 of file data_point_test.cpp.

7.111.2 Variable Documentation

7.111.2.1 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.

7.112 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, shouldNotBeAbleToCreateFromEmtyMatrix)
- TEST (DataSet, shouldNotBeAbleToCreateFromJaggedMatrix)
- TEST (DataSet, shouldNotBeAbleToCreateFromInvalidData)

7.112.1 Macro Definition Documentation

7.112.1.1 EXPECT_LONG_DOUBLE_EQ

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

7.112.2 Function Documentation

7.112.2.1 TEST() [1/4]

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

Definition at line 17 of file data_set_test.cpp.

Here is the call graph for this function:

7.112.2.2 TEST() [2/4]

```
TEST (
            DataSet ,
            shouldNotBeAbleToCreateFromEmtyMatrix  )
```

Definition at line 68 of file data_set_test.cpp.

Here is the call graph for this function:

7.112.2.3 TEST() [3/4]

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

Definition at line 108 of file data_set_test.cpp.

Here is the call graph for this function:

7.112.2.4 TEST() [4/4]

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

Definition at line 80 of file data_set_test.cpp.

Here is the call graph for this function:

7.113 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)

7.113.1 Function Documentation

7.113.1.1 TEST() [1/3]

```
TEST (
            directoryListing ,
            shouldFindFiles  )
```

Definition at line 13 of file directory_listing_test.cpp.

Here is the call graph for this function:

7.113.1.2 TEST() [2/3]

```
TEST (
            directoryListing ,
            shouldFindFilesWithDotAndDotDot  )
```

Definition at line 28 of file directory_listing_test.cpp.

Here is the call graph for this function:

7.113.1.3 TEST() [3/3]

```
TEST (
            directoryListing ,
            shouldReturnErrorWhenPathDoesNotExist  )
```

Definition at line 46 of file directory_listing_test.cpp.

Here is the call graph for this function:

7.114 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, shouldThrowExceptionWhenRaiseIsCalled)
- TEST (error, shouldCreateExpectedWithUnexpected)

Variables

- const cl::Error error

7.114.1 Function Documentation

7.114.1.1 TEST() [1/4]

```
TEST (
            error ,
            shouldCreateExpectedWithUnexpected  )
```

Definition at line 59 of file error_test.cpp.

7.114.1.2 TEST() [2/4]

```
TEST (
            error ,
            shouldPrint  )
```

Definition at line 19 of file error_test.cpp.

7.114.1.3 TEST() [3/4]

```
TEST (
            error ,
            shouldReturnValues  )
```

Definition at line 29 of file error_test.cpp.

7.114.1.4 TEST() [4/4]

```
TEST (
            error ,
            shouldThrowExceptionWhenRaiseIsCalled  )
```

Definition at line 37 of file error_test.cpp.

7.114.2 Variable Documentation

7.114.2.1 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.

7.115 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)

7.115.1 Function Documentation

7.115.1.1 TEST()

```
TEST (
            exception ,
            shouldWork  )
```

Definition at line 5 of file exception_test.cpp.

7.116 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)

7.116.1 Function Documentation

7.116.1.1 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:

7.116.1.2 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:

7.117 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)

7.117.1 Function Documentation

7.117.1.1 TEST() [1/3]

```
TEST (
            s2n ,
            shouldReturnInvalidArgumentErrorIfInputIsInvalid  )
```

Definition at line 21 of file s2n_test.cpp.

7.117.1.2 TEST() [2/3]

```
TEST (
            s2n ,
            shouldReturnOutOfRangeErrorIfInputIsOutOfRange  )
```

Definition at line 29 of file s2n_test.cpp.

7.117.1.3 TEST() [3/3]

```
TEST (
            s2n ,
            shouldWork  )
```

Definition at line 7 of file s2n_test.cpp.

Here is the call graph for this function:

7.118 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, shouldPrintCorretly)

7.118.1 Function Documentation

7.118.1.1 TEST() [1/2]

```
TEST (
            sensor ,
            shouldHaveCorrectValues  )
```

Definition at line 7 of file sensor_test.cpp.

7.118.1.2 TEST() [2/2]

```
TEST (
            sensor ,
            shouldPrintCorretly  )
```

Definition at line 15 of file sensor_test.cpp.

Here is the call graph for this function:

7.119 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)

7.119.1 Function Documentation

7.119.1.1 TEST()

```
TEST (
            to_string ,
            test  )
```

Definition at line 9 of file to_string_test.cpp.

Here is the call graph for this function:

7.120 fix_csv/include/adjust_hardware_timestamp.hpp File Reference

```
#include <cstdint>
#include <string>
```

Include dependency graph for adjust_hardware_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 &nextRowHardware↩ Timestamp, std::uint64_t *overflowCount)

7.121 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)

7.122 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)

7.123 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)

7.124 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)

7.125 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)

7.126 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)

7.127 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)

7.128 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 &nextRowHardware↩ Timestamp, std::uint64_t *overflowCount)

7.129 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)

7.130 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)

7.131 fix_csv/src/delete_non_bosch_sensors.cpp File Reference

```
#include <tl/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)

7.131.1 Macro Definition Documentation

7.131.1.1 CL_SENSOR_X

```
#define CL_SENSOR_X(
            enm,
            value ) case cl::Sensor::enm:  return PL_STRINGIFY(value);
```

7.132 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)

7.133 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)

7.134 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)

7.135 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)

7.136 fix_csv/test/adjust_hardware_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_hardware_timestamp.hpp"
```

Include dependency graph for adjust_hardware_timestamp_test.cpp:

Functions

- TEST (adjustHardwareTimestamp, shouldDoNothingForNonOverflowedValue)
- TEST (adjustHardwareTimestamp, shouldIncrementOverflowCount)
- TEST (adjustHardwareTimestamp, shouldWorkForOneRoundOfOverflow)
- TEST (adjustHardwareTimestamp, shouldWorkForTwoRoundsOfOverflow)
- TEST (adjustHardwareTimestamp, shouldWork)

7.136.1 Function Documentation

7.136.1.1 TEST() [1/5]

```
TEST (
            adjustHardwareTimestamp ,
            shouldDoNothingForNonOverflowedValue  )
```

Definition at line 15 of file adjust_hardware_timestamp_test.cpp.

Here is the call graph for this function:

7.136.1.2 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:

7.136.1.3 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:

7.136.1.4 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:

7.136.1.5 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:

7.137 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, shouldNotRemoveIfNonZerosFollow)
- TEST (removeZerosFromField, shouldNotRemoveIfNoDot)
- TEST (removeZerosFromField, shouldDoNothingIfStringIsEmpty)
- TEST (removeZerosFromField, shouldDeleteStringIfStringIsSingleDot)
- TEST (removeZerosFromField, shouldDeleteStringIfStringIsDotAndZero)

7.137.1 Function Documentation

7.137.1.1 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:

7.137.1.2 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:

7.137.1.3 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:

7.137.1.4 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:

7.137.1.5 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:

7.137.1.6 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, 126
~Process
cl::Process, 153

above_threshold.cpp
channel, 234
channelAccessor, 235
CL_CHANNEL_X, 234
above_threshold_test.cpp
EXPECT_LONG_DOUBLE_EQ, 237
TEST, 238
aboveThreshold
ctg, 52
accelerometerAverage
cl::DataSet, 102
accelerometerMaximum
cl::DataSet, 102
accelerometerThreshold
cl, 22
AccelerometerX
cl, 13
accelerometerX
cl::DataSet, 103
AccelerometerY
cl, 13
accelerometerY
cl::DataSet, 103
AccelerometerZ
cl, 13
accelerometerZ
cl::DataSet, 104
adjust_hardware_timestamp_test.cpp
TEST, 310–312
adjustHardwareTimestamp
fmc, 57
asMilliseconds
cm::ManualSegmentationPoint, 142
averageComparisonValueCalculator
ctg, 53

base_type
cl::Exception, 116
Both
cs, 44
build
cm::Configuration::Builder, 64
cs::CsvLineBuilder, 84
Builder
cm::Configuration, 81
cm::Configuration::Builder, 64
Butterworth
cs, 43

Channel
cl, 13
channel
above_threshold.cpp, 234
cl::DataPoint, 98
data_point.cpp, 265
channel.cpp
CL_CHANNEL_X, 264
channel.hpp
CL_CHANNEL, 241
CL_CHANNEL_X, 242
channel_test.cpp
TEST, 277, 278
ChannelAccessor
cl::DataSet, 102
channelAccessor
above_threshold.cpp, 235
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 >, 72
cl::data_set_accessor< Chan >, 96
cl::DataPoint, 97
channel, 98
DataPoint, 98
fileName, 98
operator<<, 100
sensor, 99
time, 99
value, 100
cl::DataSet, 101
accelerometerAverage, 102
accelerometerMaximum, 102
accelerometerX, 103
accelerometerY, 103
accelerometerZ, 104
ChannelAccessor, 102
create, 104
extractId, 106
fileName, 106
gyroscopeAverage, 107
gyroscopeMaximum, 107
gyroscopeX, 108
gyroscopeY, 108
gyroscopeZ, 109
hardwareTimestamp, 109
rowCount, 110
size_type, 102
time, 110
trigger, 111
cl::Error, 111
CL_ERROR_KIND, 112
Error, 112
file, 113
function, 113
Kind, 112
kind, 113
line, 114
message, 114
operator<<, 115
raise, 114
to_string, 115
cl::Exception, 115
base_type, 116
Exception, 117
file, 117
function, 117
line, 118
cl::fs, 24
directoryListing, 25
DirectoryListingOption, 25
ExcludeDotAndDotDot, 25
formatError, 26
None, 25
operator<, 27
operator<<, 27
operator==, 27
utf16ToUtf8, 28
utf8ToUtf16, 28
cl::fs::File, 118
copyTo, 120
create, 120
exists, 121
File, 119
moveTo, 122
path, 122
remove, 123
size, 123
cl::fs::FileStream, 124
~FileStream, 126
create, 126
FileStream, 126
OpenMode, 125
operator=, 127
PL_NONCOPYABLE, 128
Read, 126
readAll, 128
ReadWrite, 126
this_type, 125
Write, 126
write, 128
cl::fs::Path, 146
exists, 148
isDirectory, 148
isFile, 149
operator<, 151
operator<<, 151
operator==, 151
Path, 147
str, 150
cl::Process, 152
~Process, 153
create, 153
file, 154
operator=, 154
PL_NONCOPYABLE, 154
Process, 153
this_type, 153
CL_CHANNEL

channel.hpp, 241
cl, 13
CL_CHANNEL_X
above_threshold.cpp, 234
channel.cpp, 264
channel.hpp, 242
cl, 13
CL_ERROR_KIND
cl::Error, 112
error.hpp, 248
CL_ERROR_KIND_X
error.cpp, 269
error.hpp, 249
CL_FS_SEPARATOR
separator.hpp, 255
CL_SENSOR
cl, 14
sensor.hpp, 261
CL_SENSOR_X
cl, 14
delete_non_bosch_sensors.cpp, 307
sensor.cpp, 275
sensor.hpp, 261
CL_SPECIALIZE_COL_TRAITS
cl, 14–16
column.hpp, 244
CL_THROW
exception.hpp, 250
CL_THROW_FMT
exception.hpp, 250
CL_UNEXPECTED
error.hpp, 249
cm, 29
CM_DATA_SET_IDENTIFIER, 30
CM_DATA_SET_IDENTIFIER_X, 30
CM_IMU, 31
CM_IMU_X, 31
createSegmentationResults, 31
DataSetIdentifier, 30
Imu, 31
imuCount, 41
imus, 41
interpolatedDataSetPaths, 32
operator!=, 33
operator<<, 34, 35
operator==, 36
pythonOutput, 36
segment, 38
splitString, 40
toDataSetIdentifier, 40
cm::Configuration, 73
Builder, 81
deleteTooClose, 74
deleteTooCloseOptions, 74
deleteTooLowVariance, 75
deleteTooLowVarianceOptions, 75
filterKind, 76
filterKindOptions, 76
imu, 77
imuOptions, 77
operator!=, 81
operator<<, 82
operator==, 82
segmentationKind, 78
segmentationKindOptions, 78
skipWindow, 79
skipWindowOptions, 79
std::hash< Configuration >, 82
windowSize, 80
windowSizeOptions, 80
cm::Configuration::Builder, 63
build, 64
Builder, 64
deleteTooClose, 65
deleteTooLowVariance, 66
filterKind, 67
imu, 68
segmentationKind, 69
skipWindow, 70
windowSize, 71
cm::ManualSegmentationPoint, 140
asMilliseconds, 142
frame, 142
hour, 142
ManualSegmentationPoint, 141
minute, 143
operator!=, 145
operator<<, 145
operator==, 146
readCsvFile, 143
second, 144
CM_CONTAINS
configuration.cpp, 212
CM_DATA_SET_IDENTIFIER
cm, 30
data_set_identifier.hpp, 204
CM_DATA_SET_IDENTIFIER_X
cm, 30
data_set_identifier.cpp, 214
data_set_identifier.hpp, 204
CM_DEV_NULL
python_output.cpp, 218
CM_ENSURE_CONTAINS
configuration.cpp, 212
CM_ENSURE_HAS_VALUE
configuration.cpp, 212
CM_IMU
cm, 31
imu.hpp, 205
CM_IMU_X
cm, 31
imu.cpp, 215
imu.hpp, 206
CM_SEGMENTOR
python_output.cpp, 218
CMakeLists.txt

include, 155–159
set, 155–159
Column
cl, 13
column.hpp
CL_SPECIALIZE_COL_TRAITS, 244
column_index
cl, 23
column_test.cpp
TEST, 279
column_type
cl, 12
compare_segmentation/CMakeLists.txt, 155
compare_segmentation/include/csv_line.hpp, 160
compare_segmentation/include/data_set_info.hpp, 161
compare_segmentation/include/filter_kind.hpp, 163
compare_segmentation/include/log_files.hpp, 164
compare_segmentation/include/log_info.hpp, 164
compare_segmentation/include/log_line.hpp, 165
compare_segmentation/include/paths.hpp, 166
compare_segmentation/include/segmentation_kind.hpp, 167
compare_segmentation/src/csv_line.cpp, 168
compare_segmentation/src/data_set_info.cpp, 169
compare_segmentation/src/filter_kind.cpp, 169
compare_segmentation/src/log_files.cpp, 170
compare_segmentation/src/log_info.cpp, 171
compare_segmentation/src/log_line.cpp, 172
compare_segmentation/src/main.cpp, 172
compare_segmentation/src/segmentation_kind.cpp, 183
compare_segmentation/test/CMakeLists.txt, 155
compare_segmentation/test/csv_line_test.cpp, 184
compare_segmentation/test/data_set_info_test.cpp, 185
compare_segmentation/test/log_files_test.cpp, 186
compare_segmentation/test/log_info_test.cpp, 188
compare_segmentation/test/log_line_test.cpp, 198
compare_segmentation/test/main.cpp, 174
configuration.cpp
CM_CONTAINS, 212
CM_ENSURE_CONTAINS, 212
CM_ENSURE_HAS_VALUE, 212
confusion_matrix/CMakeLists.txt, 159
confusion_matrix/include/configuration.hpp, 200
confusion_matrix/include/create_segmentation_results.hpp, 201
confusion_matrix/include/data_set_identifier.hpp, 202
confusion_matrix/include/imu.hpp, 204
confusion_matrix/include/interpolated_data_set_paths.hpp, 207
confusion_matrix/include/manual_segmentation_point.hpp, 208
confusion_matrix/include/python_output.hpp, 209
confusion_matrix/include/segment.hpp, 209
confusion_matrix/include/split_string.hpp, 210
confusion_matrix/src/configuration.cpp, 211
confusion_matrix/src/create_segmentation_results.cpp, 213
confusion_matrix/src/data_set_identifier.cpp, 213
confusion_matrix/src/imu.cpp, 215
confusion_matrix/src/interpolated_data_set_paths.cpp, 216
confusion_matrix/src/main.cpp, 181
confusion_matrix/src/manual_segmentation_point.cpp, 216
confusion_matrix/src/python_output.cpp, 217
confusion_matrix/src/segment.cpp, 218
confusion_matrix/src/split_string.cpp, 219
confusion_matrix/test/CMakeLists.txt, 159
confusion_matrix/test/data_set_identifier_test.cpp, 219
confusion_matrix/test/interpolated_data_set_paths_test.cpp, 221
confusion_matrix/test/main.cpp, 182
confusion_matrix/test/manual_segmentation_point_test.cpp, 222
confusion_matrix/test/segment_test.cpp, 225
confusion_matrix/test/split_string_test.cpp, 227
convertToUnixLineEndings
fmc, 57
copyTo
cl::fs::File, 120
counting/CMakeLists.txt, 156
counting/include/above_threshold.hpp, 228
counting/include/average_comparison_value_calculator.hpp, 229
counting/include/half_maximum_comparison_value_calculator.hpp, 229
counting/include/is_relevant.hpp, 230
counting/include/percentage_of.hpp, 231
counting/include/run_above_threshold.hpp, 232
counting/src/above_threshold.cpp, 233
counting/src/average_comparison_value_calculator.cpp, 235
counting/src/half_maximum_comparison_value_calculator.cpp, 236
counting/src/main.cpp, 175
counting/src/run_above_threshold.cpp, 236
counting/test/above_threshold_test.cpp, 237
counting/test/CMakeLists.txt, 156
counting/test/main.cpp, 177
counting/test/percentage_of_test.cpp, 239
create
cl::DataSet, 104
cl::fs::File, 120
cl::fs::FileStream, 126
cl::Process, 153
cs::LogInfo, 131
createBackupFile
fmc, 58
createSegmentationResults
cm, 31
cs, 42
Both, 44
Butterworth, 43

CS_SPECIALIZE_DATA_SET_INFO, 44–47
FilterKind, 43
logFiles, 47
logPath, 51
Maxima, 44
Minima, 44
MovingAverage, 43
oldLogPath, 51
operator!=, 48
operator<<, 49
operator==, 50
PL_DEFINE_EXCEPTION_TYPE, 50
repetitionCount, 50
SegmentationKind, 43
cs::CsvLineBuilder, 83
build, 84
CsvLineBuilder, 84
dataSet, 85
deleteLowVariance, 86
deleteTooClose, 87
filter, 88
isOld, 89
kind, 90
repetitions, 91
segmentationPoints, 92
sensor, 93
skipWindow, 94
this_type, 84
windowSize, 95
cs::data_set_info< Tag >, 97
cs::LogInfo, 130
create, 131
deleteLowVariance, 132
deleteTooClose, 132
filterKind, 132
invalidSensor, 136
isInitialized, 133
logFilePath, 133
LogInfo, 131
operator!=, 135
operator<<, 135
operator==, 136
segmentationKind, 133
sensor, 134
skipWindow, 134
windowSize, 134
cs::LogLine, 136
fileName, 137
filePath, 138
invalidSensor, 140
parse, 138
segmentationPointCount, 139
sensor, 139
CS_SPECIALIZE_DATA_SET_INFO
cs, 44–47
data_set_info.hpp, 162
csv_lib/CMakeLists.txt, 157
csv_lib/include/cl/channel.hpp, 240
csv_lib/include/cl/column.hpp, 243
csv_lib/include/cl/data_point.hpp, 245
csv_lib/include/cl/data_set.hpp, 246
csv_lib/include/cl/dos2unix.hpp, 247
csv_lib/include/cl/error.hpp, 248
csv_lib/include/cl/exception.hpp, 249
csv_lib/include/cl/fs/directory_listing.hpp, 251
csv_lib/include/cl/fs/file.hpp, 252
csv_lib/include/cl/fs/file_stream.hpp, 253
csv_lib/include/cl/fs/path.hpp, 254
csv_lib/include/cl/fs/separator.hpp, 254
csv_lib/include/cl/fs/windows.hpp, 256
csv_lib/include/cl/process.hpp, 257
csv_lib/include/cl/read_csv_file.hpp, 258
csv_lib/include/cl/s2n.hpp, 259
csv_lib/include/cl/sensor.hpp, 259
csv_lib/include/cl/to_string.hpp, 262
csv_lib/include/cl/use_unbuffered_io.hpp, 263
csv_lib/src/cl/channel.cpp, 263
csv_lib/src/cl/data_point.cpp, 265
csv_lib/src/cl/data_set.cpp, 268
csv_lib/src/cl/dos2unix.cpp, 268
csv_lib/src/cl/error.cpp, 269
csv_lib/src/cl/exception.cpp, 270
csv_lib/src/cl/fs/directory_listing.cpp, 270
csv_lib/src/cl/fs/file.cpp, 271
csv_lib/src/cl/fs/file_stream.cpp, 271
csv_lib/src/cl/fs/path.cpp, 272
csv_lib/src/cl/fs/windows.cpp, 273
csv_lib/src/cl/process.cpp, 273
csv_lib/src/cl/read_csv_file.cpp, 274
csv_lib/src/cl/sensor.cpp, 275
csv_lib/src/cl/use_unbuffered_io.cpp, 276
csv_lib/test/channel_test.cpp, 276
csv_lib/test/CMakeLists.txt, 157
csv_lib/test/column_test.cpp, 278
csv_lib/test/data_point_test.cpp, 280
csv_lib/test/data_set_test.cpp, 282
csv_lib/test/directory_listing_test.cpp, 286
csv_lib/test/error_test.cpp, 288
csv_lib/test/exception_test.cpp, 290
csv_lib/test/main.cpp, 178
csv_lib/test/read_csv_file_test.cpp, 291
csv_lib/test/s2n_test.cpp, 292
csv_lib/test/sensor_test.cpp, 294
csv_lib/test/to_string_test.cpp, 295
csv_line_test.cpp
TEST, 185
CsvFileKind
cl, 13
CsvLineBuilder
cs::CsvLineBuilder, 84
ctg, 52
aboveThreshold, 52
averageComparisonValueCalculator, 53
halfMaximumComparisonValueCalculator, 53
isRelevant, 54
percentageOf, 55

runAboveThreshold, 55

data_point.cpp
channel, 265
fileName, 265
sensor, 266
time, 266
value, 267
data_point_test.cpp
dp, 281
TEST, 280, 281
data_set_accessor_v
cl, 23
data_set_identifier.cpp
CM_DATA_SET_IDENTIFIER_X, 214
DSI, 214
data_set_identifier.hpp
CM_DATA_SET_IDENTIFIER, 204
CM_DATA_SET_IDENTIFIER_X, 204
data_set_identifier_test.cpp
DSI, 220
TEST, 220
data_set_info.hpp
CS_SPECIALIZE_DATA_SET_INFO, 162
data_set_info_test.cpp
TEST, 185
data_set_test.cpp
EXPECT_LONG_DOUBLE_EQ, 282
TEST, 282–285
DataPoint
cl::DataPoint, 98
dataSet
cs::CsvLineBuilder, 85
dataSetAccessor
cl, 16
DataSetIdentifier
cm, 30
delete_non_bosch_sensors.cpp
CL_SENSOR_X, 307
deleteLowVariance
cs::CsvLineBuilder, 86
cs::LogInfo, 132
deleteNonBoschSensors
fmc, 58
deleteOutOfBoundsValues
fmc, 59
deleteTooClose
cm::Configuration, 74
cm::Configuration::Builder, 65
cs::CsvLineBuilder, 87
cs::LogInfo, 132
deleteTooCloseOptions
cm::Configuration, 74
deleteTooLowVariance
cm::Configuration, 75
cm::Configuration::Builder, 66
deleteTooLowVarianceOptions
cm::Configuration, 75
directory_listing_test.cpp
TEST, 287
directoryListing
cl::fs, 25
DirectoryListingOption
cl::fs, 25
dos2unix
cl, 16
dp
data_point_test.cpp, 281
DSI
data_set_identifier.cpp, 214
data_set_identifier_test.cpp, 220
manual_segmentation_point.cpp, 217
manual_segmentation_point_test.cpp, 222

Error
cl::Error, 112
error
error_test.cpp, 289
error.cpp
CL_ERROR_KIND_X, 269
error.hpp
CL_ERROR_KIND, 248
CL_ERROR_KIND_X, 249
CL_UNEXPECTED, 249
error_test.cpp
error, 289
TEST, 289
Exception
cl::Exception, 117
exception.hpp
CL_THROW, 250
CL_THROW_FMT, 250
exception_test.cpp
TEST, 290
ExcludeDotAndDotDot
cl::fs, 25
exists
cl::fs::File, 121
cl::fs::Path, 148
EXPECT_LONG_DOUBLE_EQ
above_threshold_test.cpp, 237
data_set_test.cpp, 282
percentage_of_test.cpp, 239
EXPECT_SEGMENTATION_POINTS
segment_test.cpp, 226
Expected
cl, 12
ExtractId
cl, 13
extractId
cl::DataSet, 106

File
cl::fs::File, 119
file
cl::Error, 113
cl::Exception, 117
cl::Process, 154

fileName
cl::DataPoint, 98
cl::DataSet, 106
cs::LogLine, 137
data_point.cpp, 265
filePath
cs::LogLine, 138
FileStream
cl::fs::FileStream, 126
filter
cs::CsvLineBuilder, 88
FilterKind
cs, 43
filterKind
cm::Configuration, 76
cm::Configuration::Builder, 67
cs::LogInfo, 132
filterKindOptions
cm::Configuration, 76
fix_csv/CMakeLists.txt, 158
fix_csv/include/adjust_hardware_timestamp.hpp, 296
fix_csv/include/convert_to_unix_line_endings.hpp, 297
fix_csv/include/create_backup_file.hpp, 298
fix_csv/include/delete_non_bosch_sensors.hpp, 299
fix_csv/include/delete_out_of_bounds_values.hpp, 300
fix_csv/include/remove_zeros_from_field.hpp, 301
fix_csv/include/restore_from_backup.hpp, 302
fix_csv/include/write_file.hpp, 303
fix_csv/src/adjust_hardware_timestamp.cpp, 304
fix_csv/src/convert_to_unix_line_endings.cpp, 305
fix_csv/src/create_backup_file.cpp, 306
fix_csv/src/delete_non_bosch_sensors.cpp, 306
fix_csv/src/delete_out_of_bounds_values.cpp, 307
fix_csv/src/main.cpp, 179
fix_csv/src/remove_zeros_from_field.cpp, 308
fix_csv/src/restore_from_backup.cpp, 308
fix_csv/src/write_file.cpp, 309
fix_csv/test/adjust_hardware_timestamp_test.cpp, 310
fix_csv/test/CMakeLists.txt, 158
fix_csv/test/main.cpp, 180
fix_csv/test/remove_zeros_from_field_test.cpp, 312
Fixed
cl, 14
fmc, 56
adjustHardwareTimestamp, 57
convertToUnixLineEndings, 57
createBackupFile, 58
deleteNonBoschSensors, 58
deleteOutOfBoundsValues, 59
removeZerosFromField, 59
restoreFromBackup, 60
writeFile, 60
formatError
cl::fs, 26
frame
cm::ManualSegmentationPoint, 142
function
cl::Error, 113
cl::Exception, 117

gyroscopeAverage
cl::DataSet, 107
gyroscopeMaximum
cl::DataSet, 107
gyroscopeThreshold
cl, 23
GyroscopeX
cl, 13
gyroscopeX
cl::DataSet, 108
GyroscopeY
cl, 13
gyroscopeY
cl::DataSet, 108
GyroscopeZ
cl, 13
gyroscopeZ
cl::DataSet, 109

halfMaximumComparisonValueCalculator
ctg, 53
HardwareTimestamp
cl, 13
hardwareTimestamp
cl::DataSet, 109
hour
cm::ManualSegmentationPoint, 142

Imu
cm, 31
imu
cm::Configuration, 77
cm::Configuration::Builder, 68
imu.cpp
CM_IMU_X, 215
imu.hpp
CM_IMU, 205
CM_IMU_X, 206
imuCount
cm, 41
imuOptions
cm::Configuration, 77
imus
cm, 41
include
CMakeLists.txt, 155–159
interpolated_data_set_paths_test.cpp
TEST, 221
interpolatedDataSetPaths
cm, 32
invalidSensor
cs::LogInfo, 136
cs::LogLine, 140
isAccelerometer
cl, 17
isDirectory
cl::fs::Path, 148

isFile
cl::fs::Path, 149
isGyroscope
cl, 17
isInitialized
cs::LogInfo, 133
isOld
cs::CsvLineBuilder, 89
isRelevant
ctg, 54

Kind
cl::Error, 112
kind
cl::Error, 113
cs::CsvLineBuilder, 90

line
cl::Error, 114
cl::Exception, 118
log_files_test.cpp
TEST, 186, 187
log_info_test.cpp
TEST, 189–197
log_line_test.cpp
TEST, 198–200
logFilePath
cs::LogInfo, 133
logFiles
cs, 47
LogInfo
cs::LogInfo, 131
logPath
cs, 51

main
main.cpp, 173, 175, 176, 178–181, 183
main.cpp
main, 173, 175, 176, 178–181, 183
manual_segmentation_point.cpp
DSI, 217
manual_segmentation_point_test.cpp
DSI, 222
TEST, 223–225
ManualSegmentationPoint
cm::ManualSegmentationPoint, 141
Maxima
cs, 44
message
cl::Error, 114
Minima
cs, 44
minute
cm::ManualSegmentationPoint, 143
moveTo
cl::fs::File, 122
MovingAverage
cs, 43

None
cl::fs, 25

oldLogPath
cs, 51
OpenMode
cl::fs::FileStream, 125
operator!=
cm, 33
cm::Configuration, 81
cm::ManualSegmentationPoint, 145
cs, 48
cs::LogInfo, 135
operator<
cl::fs, 27
cl::fs::Path, 151
operator<<
cl, 18, 19
cl::DataPoint, 100
cl::Error, 115
cl::fs, 27
cl::fs::Path, 151
cm, 34, 35
cm::Configuration, 82
cm::ManualSegmentationPoint, 145
cs, 49
cs::LogInfo, 135
operator()
std::hash<::cl::fs::Path >, 129
std::hash<::cm::Configuration >, 129
operator=
cl::fs::FileStream, 127
cl::Process, 154
operator==
cl::fs, 27
cl::fs::Path, 151
cm, 36
cm::Configuration, 82
cm::ManualSegmentationPoint, 146
cs, 50
cs::LogInfo, 136

parse
cs::LogLine, 138
Path
cl::fs::Path, 147
path
cl::fs::File, 122
percentage_of_test.cpp
EXPECT_LONG_DOUBLE_EQ, 239
TEST, 239
percentageOf
ctg, 55
PL_DEFINE_EXCEPTION_TYPE
cs, 50
PL_NONCOPYABLE
cl::fs::FileStream, 128
cl::Process, 154
Process

cl::Process, 153
python_output.cpp
CM_DEV_NULL, 218
CM_SEGMENTOR, 218
pythonOutput
cm, 36

raise
cl::Error, 114
Raw
cl, 14
Read
cl::fs::FileStream, 126
read_csv_file_test.cpp
TEST, 291, 292
readAll
cl::fs::FileStream, 128
readCsvFile
cl, 20
cm::ManualSegmentationPoint, 143
ReadWrite
cl::fs::FileStream, 126
remove
cl::fs::File, 123
remove_zeros_from_field_test.cpp
TEST, 313–315
removeZerosFromField
fmc, 59
repetitionCount
cs, 50
repetitions
cs::CsvLineBuilder, 91
restoreFromBackup
fmc, 60
rowCount
cl::DataSet, 110
runAboveThreshold
ctg, 55

s2n
cl, 20
s2n_test.cpp
TEST, 293
SamplingRate
cl, 13
second
cm::ManualSegmentationPoint, 144
segment
cm, 38
segment_test.cpp
EXPECT_SEGMENTATION_POINTS, 226
TEST, 226
SegmentationKind
cs, 43
segmentationKind
cm::Configuration, 78
cm::Configuration::Builder, 69
cs::LogInfo, 133
segmentationKindOptions
cm::Configuration, 78
segmentationPointCount
cs::LogLine, 139
segmentationPoints
cs::CsvLineBuilder, 92
Sensor
cl, 14
sensor
cl::DataPoint, 99
cs::CsvLineBuilder, 93
cs::LogInfo, 134
cs::LogLine, 139
data_point.cpp, 266
sensor.cpp
CL_SENSOR_X, 275
sensor.hpp
CL_SENSOR, 261
CL_SENSOR_X, 261
sensor_test.cpp
TEST, 295
sensors
cl, 24
separator.hpp
CL_FS_SEPARATOR, 255
set
CMakeLists.txt, 155–159
size
cl::fs::File, 123
size_type
cl::DataSet, 102
skipWindow
cm::Configuration, 79
cm::Configuration::Builder, 70
cs::CsvLineBuilder, 94
cs::LogInfo, 134
skipWindowOptions
cm::Configuration, 79
split_string_test.cpp
TEST, 227
splitString
cm, 40
std::hash< Configuration >
cm::Configuration, 82
std::hash<::cl::fs::Path >, 129
operator(), 129
std::hash<::cm::Configuration >, 129
operator(), 129
str
cl::fs::Path, 150

TEST
above_threshold_test.cpp, 238
adjust_hardware_timestamp_test.cpp, 310–312
channel_test.cpp, 277, 278
column_test.cpp, 279
csv_line_test.cpp, 185
data_point_test.cpp, 280, 281
data_set_identifier_test.cpp, 220
data_set_info_test.cpp, 185

data_set_test.cpp, 282–285
directory_listing_test.cpp, 287
error_test.cpp, 289
exception_test.cpp, 290
interpolated_data_set_paths_test.cpp, 221
log_files_test.cpp, 186, 187
log_info_test.cpp, 189–197
log_line_test.cpp, 198–200
manual_segmentation_point_test.cpp, 223–225
percentage_of_test.cpp, 239
read_csv_file_test.cpp, 291, 292
remove_zeros_from_field_test.cpp, 313–315
s2n_test.cpp, 293
segment_test.cpp, 226
sensor_test.cpp, 295
split_string_test.cpp, 227
to_string_test.cpp, 296
this_type
cl::fs::FileStream, 125
cl::Process, 153
cs::CsvLineBuilder, 84
threshold
cl, 21
Time
cl, 13
time
cl::DataPoint, 99
cl::DataSet, 110
data_point.cpp, 266
to_string
cl, 21
cl::Error, 115
to_string_test.cpp
TEST, 296
toDataSetIdentifier
cm, 40
Trigger
cl, 13
trigger
cl::DataSet, 111

useUnbufferedIo
cl, 22
utf16ToUtf8
cl::fs, 28
utf8ToUtf16
cl::fs, 28

value
cl::DataPoint, 100
data_point.cpp, 267

windowSize
cm::Configuration, 80
cm::Configuration::Builder, 71
cs::CsvLineBuilder, 95
cs::LogInfo, 134
windowSizeOptions
cm::Configuration, 80
Write
cl::fs::FileStream, 126
write
cl::fs::FileStream, 128
writeFile
fmc, 60