Hobby-Eberly Telescope Control System

3.12277∗

Generated by Doxygen 1.6.1

Tue Jan 19 19:46:52 2021

Contents

1 Main Page **1**

1.1 HET Processes 1

1.2 HET Inter-Process Communication 2

1.3 HET Support Libraries 2

2 Module Index **5**

2.1 Modules 5

3 Namespace Index **9**

3.1 Namespace List 9

4 Class Index **11**

4.1 Class Hierarchy 11

5 Class Index **35**

5.1 Class List 35

6 File Index **57**

6.1 File List 57

7 Module Documentation **65**

7.1 Configuration 65

7.1.1 Detailed Description 65

7.2 Outlet 66

7.3 Apc 67

7.4 Image pdu Sub-System 68

7.5 Image Pdu 69

7.6 Payload Alignment System (APC) 70

7.6.1 Detailed Description 70

7.7 APC Message Handlers 71

7.8 APC Utility Classes . . . 72
7.9 CAMRA Configuration Classes . . . 73
7.9.1 Detailed Description . . . 73
7.10 CAMRA Handlers . . . 74
7.11 CAMRA Hardware Classes . . . 75
7.12 Array sizing constants . . . 76
7.12.1 Detailed Description . . . 76
7.13 Controller initialization constants . . . 77
7.13.1 Detailed Description . . . 77
7.14 Amplifier related constants . . . 78
7.14.1 Detailed Description . . . 78
7.15 CCD Array Management and Recording Application (CAMRA) . . . 79
7.15.1 Detailed Description . . . 79
7.16 CAMRA Classes . . . 80
7.17 TCS Fixed precision time . . . 81
7.18 HET Earth orientation prediction . . . 82
7.19 TCS Application Class . . . 83
7.19.1 Detailed Description . . . 83
7.20 TCS Base Class . . . 84
7.21 Common Library . . . 85
7.22 TCS Base Configuration . . . 86
7.23 Coordinate Systems and Transformations . . . 87
7.23.1 Detailed Description . . . 87
7.24 TCS SubSystem Class . . . 88
7.25 Coordinate Transforms and Structure Mount Model . . . 89
7.26 Coordinate Transform Library . . . 90
7.27 Structure Mount Model . . . 91
7.28 Common Client-Side Logging . . . 92
7.29 Remote signalling . . . 93
7.29.1 Detailed Description . . . 93
7.29.2 Define Documentation . . . 93
7.29.2.1 MESSAGE_SETJMP . . . 93
7.30 Messaging Library . . . 94
7.30.1 Detailed Description . . . 94
7.31 TinyThread++ API Reference . . . 95
7.31.1 Introduction . . . 95

7.31.2 Portability . . . 95
7.31.3 Classes . . . 95
7.31.4 Miscellaneous . . . 95
7.32 Configuration . . . 96
7.32.1 Detailed Description . . . 96
7.33 Dome . . . 97
7.34 Dome . . . 98
7.35 Structure . . . 99
7.36 Weather . . . 100
7.37 Dome . . . 101
7.38 Dome . . . 102
7.39 Legacy Servers Access System (Legacy) . . . 103
7.39.1 Detailed Description . . . 103
7.40 Class Documentation . . . 104
7.41 Shutter . . . 105
7.42 Shutter . . . 106
7.43 Structure . . . 107
7.44 Structure . . . 108
7.45 Weather . . . 109
7.46 Weather . . . 110
7.47 Camera . . . 111
7.48 AVT Camera . . . 112
7.49 FLI Camera . . . 113
7.50 PointGrey Camera . . . 114
7.51 Simulated Camera . . . 115
7.52 Image Processor . . . 116
7.53 Light Source . . . 117
7.54 Configuration . . . 118
7.54.1 Detailed Description . . . 118
7.55 Light Source . . . 119
7.56 Image Processor . . . 120
7.57 Pipe Line . . . 121
7.58 Image Storage . . . 122
7.59 Image Pipeline Sub-System . . . 123
7.60 Image Pipeline . . . 124
7.61 Photometry Image Processor . . . 125

7.62 Wave Front Sensor Image Processor . . . 126
7.63 FITS Image Storage . . . 127
7.64 Payload Alignment System (PAS) . . . 128
7.64.1 Detailed Description . . . 128
7.65 PAS Message Handlers . . . 129
7.66 PAS Utility Classes . . . 130
7.67 Tip Tilt Light Source . . . 131
7.68 Acquisition and Guide Probe Assembly . . . 132
7.68.1 Detailed Description . . . 132
7.69 FCUClasses . . . 133
7.70 MODBUS Utility Data Structures . . . 134
7.71 Prime Focus Instrument Package Server (PFIPServer) . . . 135
7.71.1 Detailed Description . . . 135
7.72 PFIP Configuration Settings . . . 137
7.73 PFIPServer Classes . . . 138
7.74 PFIP Interface Operations and Events . . . 139
7.75 Acquisition and Probe Assembly . . . 140
7.76 Facility Calibration Unit . . . 141
7.77 Probe Assembly classes . . . 142
7.77.1 Detailed Description . . . 142
7.77.2 Enumeration Type Documentation . . . 142
7.77.2.1 WhichMotor . . . 142
7.78 EPICSVariables . . . 143
7.79 Configuration . . . 144
7.79.1 Detailed Description . . . 144
7.80 Sample Servers Access System (Sample) . . . 145
7.80.1 Detailed Description . . . 145
7.81 Message Handler Documentation . . . 146
7.82 Class Documentation . . . 147
7.83 Configuration . . . 148
7.83.1 Detailed Description . . . 148
7.84 Pointing Servers Access System (Pointing) . . . 149
7.84.1 Detailed Description . . . 149
7.85 Class Documentation . . . 150
7.86 Telescope Control System (TCS) . . . 151
7.87 Tracker Motion Control System (TMCS) API . . . 152

7.87.1 Detailed Description . . . 152
7.87.2 SET_TRAJ . . . 152
7.87.2.1 Reference . . . 152
7.87.2.2 Example . . . 153
7.87.2.3 Returns . . . 153
7.87.2.4 Notes . . . 153
7.87.3 ABORT . . . 153
7.87.3.1 Reference . . . 154
7.87.3.2 Example . . . 154
7.87.3.3 Returns . . . 154
7.87.3.4 Notes . . . 154
7.87.4 MOVR_RTF . . . 154
7.87.4.1 Reference . . . 155
7.87.4.2 Example . . . 155
7.87.4.3 Returns . . . 155
7.87.4.4 Notes . . . 155
7.87.5 MOVA_RTF . . . 156
7.87.5.1 Reference . . . 156
7.87.5.2 Example . . . 156
7.87.5.3 Returns . . . 156
7.87.5.4 Notes . . . 157
7.87.6 MOVR_TSF . . . 157
7.87.6.1 Reference . . . 158
7.87.6.2 Example . . . 158
7.87.6.3 Returns . . . 158
7.87.6.4 Reference . . . 158
7.87.7 MOVA_TSF . . . 159
7.87.7.1 Reference . . . 159
7.87.7.2 Example . . . 160
7.87.7.3 Returns . . . 160
7.87.7.4 Notes . . . 160
7.87.8 INIT . . . 161
7.87.8.1 Example . . . 161
7.87.8.2 Returns . . . 161
7.87.8.3 Notes . . . 162
7.87.9 Watchdog Command . . . 162

7.87.9.1 Example . . . 162
7.87.9.2 Returns . . . 162
7.87.9.3 Notes . . . 162
7.87.10 MSG . . . 163
7.87.10.1 Reference . . . 164
7.87.10.2 Format . . . 164
7.87.10.3 Returns . . . 164
7.87.10.4 tmcs_Notes . . . 165
7.87.11 CLR . . . 165
7.87.11.1 Reference . . . 165
7.87.11.2 Format . . . 165
7.87.11.3 Example . . . 165
7.87.11.4 Returns . . . 165
7.87.11.5 pNotes . . . 165
7.87.12 MOVE_TIME . . . 166
7.87.12.1 Reference . . . 166
7.87.12.2 Example . . . 166
7.87.12.3 Returns . . . 166
7.87.12.4 Notes . . . 167
7.87.13 TMCS_return_values . . . 167
7.88 Tracker Server . . . 168
7.88.1 Detailed Description . . . 168

8 Namespace Documentation **169**
8.1 equ_hor Namespace Reference . . . 169
8.1.1 Detailed Description . . . 169
8.1.2 Function Documentation . . . 169
8.1.2.1 equ2hor . . . 169
8.2 hetlib::hetutils Namespace Reference . . . 170
8.2.1 Detailed Description . . . 170
8.2.2 Function Documentation . . . 171
8.2.2.1 Att . . . 171
8.2.2.2 BestAzimuth . . . 171
8.2.2.3 clipToCircle_deg . . . 171
8.2.2.4 clipToCircle_rad . . . 171
8.2.2.5 degrees_to_hours . . . 171
8.2.2.6 Ho . . . 172

8.2.2.7 hours_to_degrees . . . 172
8.2.2.8 hours_to_radians . . . 172
8.2.2.9 modulo . . . 172
8.2.2.10 Pa . . . 172
8.2.2.11 radians_to_hours . . . 172
8.2.2.12 Tde . . . 172
8.3 hetlib::Points Namespace Reference . . . 173
8.3.1 Detailed Description . . . 173
8.3.2 Variable Documentation . . . 173
8.3.2.1 knownLists . . . 173
8.4 hetlib::tcsconfig Namespace Reference . . . 175
8.4.1 Detailed Description . . . 175
8.5 hetlib::tcsutils Namespace Reference . . . 176
8.5.1 Detailed Description . . . 176
8.5.2 Function Documentation . . . 176
8.5.2.1 start_clients . . . 176
8.6 hetutils Namespace Reference . . . 177
8.6.1 Detailed Description . . . 177
8.6.2 Function Documentation . . . 178
8.6.2.1 Att . . . 178
8.6.2.2 BestAzimuth . . . 178
8.6.2.3 clipToCircle_deg . . . 178
8.6.2.4 clipToCircle_rad . . . 178
8.6.2.5 degrees_to_hours . . . 179
8.6.2.6 Ho . . . 179
8.6.2.7 hours_to_degrees . . . 179
8.6.2.8 hours_to_radians . . . 179
8.6.2.9 modulo . . . 179
8.6.2.10 Pa . . . 179
8.6.2.11 radians_to_hours . . . 179
8.6.2.12 Tde . . . 179
8.7 Points Namespace Reference . . . 180
8.7.1 Detailed Description . . . 180
8.7.2 Variable Documentation . . . 180
8.7.2.1 knownLists . . . 180
8.8 pt_itf Namespace Reference . . . 182

8.8.1 Detailed Description . . . 182
8.8.2 Function Documentation . . . 182
8.8.2.1 find_radec . . . 182
8.8.2.2 inEllipse . . . 183
8.9 st3 Namespace Reference . . . 184
8.9.1 Detailed Description . . . 184
8.9.2 Function Documentation . . . 184
8.9.2.1 getCurrentIndexTime . . . 184
8.10 tcsconfig Namespace Reference . . . 185
8.10.1 Detailed Description . . . 185
8.11 tcsutils Namespace Reference . . . 186
8.11.1 Detailed Description . . . 186
8.11.2 Function Documentation . . . 186
8.11.2.1 getposition . . . 186
8.11.2.2 moveStructure . . . 186
8.11.2.3 named_lookup . . . 186
8.11.2.4 start_clients . . . 187
8.12 tthread Namespace Reference . . . 188
8.12.1 Detailed Description . . . 188
8.12.2 Typedef Documentation . . . 189
8.12.2.1 pthread_trampoline . . . 189
8.12.3 Function Documentation . . . 189
8.12.3.1 _threadpool_start_thread . . . 189
8.13 tthread::chrono Namespace Reference . . . 190
8.13.1 Detailed Description . . . 190
8.14 tthread::this_thread Namespace Reference . . . 191
8.14.1 Detailed Description . . . 191
8.14.2 Function Documentation . . . 191
8.14.2.1 sleep_for . . . 191
8.14.2.2 yield . . . 191

9 Class Documentation **193**

9.1 _modbus Struct Reference . . . 193
9.2 _modbus_backend Struct Reference . . . 194
9.3 _modbus_rtu Struct Reference . . . 195
9.4 _modbus_tcp Struct Reference . . . 196
9.5 _modbus_tcp_pi Struct Reference . . . 197

9.6 _sft Struct Reference . . . 198
9.7 tthread::_thread_start_functor< T > Struct Template Reference . . . 199
9.8 tthread::_thread_start_functor_arg< F, A > Struct Template Reference . . . 200
9.9 tthread::_thread_start_info Struct Reference . . . 201
9.10 tthread::_thread_start_member< 14463331 > Struct Template Reference . . . 202
9.11 tthread::_thread_start_member_arg< 14463331, A > Struct Template Reference . . . 203
9.12 Abort Class Reference . . . 204
9.12.1 Detailed Description . . . 204
9.13 camra_handlers::abort Class Reference . . . 205
9.13.1 Member Function Documentation . . . 205
9.13.1.1 process . . . 205
9.14 abort Class Reference . . . 206
9.14.1 Detailed Description . . . 206
9.15 abortMove Class Reference . . . 207
9.15.1 Detailed Description . . . 207
9.16 tracker_handlers::abortMove Class Reference . . . 208
9.16.1 Member Function Documentation . . . 208
9.16.1.1 process . . . 208
9.17 Ui::aboutDialog Class Reference . . . 209
9.18 aboutDialog Class Reference . . . 210
9.19 ACQ_Camera Class Reference . . . 211
9.19.1 Member Function Documentation . . . 212
9.19.1.1 Enable . . . 212
9.19.1.2 GetACQCameraFilterPos . . . 212
9.19.1.3 GetACQCameraStatus . . . 212
9.19.1.4 Initialize . . . 212
9.19.1.5 SetACQCameraFilter . . . 212
9.19.1.6 SetACQCameraStatus . . . 212
9.20 ACQ_Shutter Class Reference . . . 213
9.20.1 Constructor & Destructor Documentation . . . 215
9.20.1.1 ACQ_Shutter . . . 215
9.20.2 Member Function Documentation . . . 215
9.20.2.1 CloseShutter . . . 215
9.20.2.2 EnableShutter . . . 215
9.20.2.3 GetShutterStatus . . . 215
9.20.2.4 HomeShutter . . . 215

9.20.2.5 Initialize . . . 215
9.20.2.6 OpenShutter . . . 215
9.20.2.7 SetShutterStatus . . . 216
9.20.2.8 WriteShutterOpenClose . . . 216
9.21 ACQ_SubSystem Class Reference . . . 217
9.21.1 Member Function Documentation . . . 220
9.21.1.1 EnableDisableProbes . . . 220
9.21.1.2 GetACQCameraFocus . . . 220
9.21.1.3 SetACQCameraFilter . . . 221
9.21.1.4 SetACQCameraFocus . . . 221
9.21.1.5 SetACQCameraFocus_mm . . . 221
9.21.1.6 SetCWFSCameraDeployed . . . 221
9.21.1.7 SetDither . . . 221
9.21.1.8 SetENWAFilterPosition . . . 221
9.21.1.9 TakeExposure . . . 221
9.22 ACQCamInfo Class Reference . . . 222
9.23 AcqCamStatus Class Reference . . . 223
9.24 Ui::acqCamStatusTab Class Reference . . . 224
9.25 acqCamStatusTabUi Class Reference . . . 225
9.26 AcqCWFSStatus Class Reference . . . 226
9.27 AcqENWAStatus Class Reference . . . 227
9.28 AcqMiscStatus Class Reference . . . 228
9.29 AcqShutterStatus Class Reference . . . 229
9.30 activate Class Reference . . . 230
9.30.1 Detailed Description . . . 230
9.30.2 Member Function Documentation . . . 230
9.30.2.1 process . . . 230
9.31 add_correction Class Reference . . . 231
9.31.1 Detailed Description . . . 231
9.31.2 Member Function Documentation . . . 231
9.31.2.1 process . . . 231
9.32 AdjustDither Class Reference . . . 232
9.32.1 Detailed Description . . . 232
9.33 Alarm_Monitor Class Reference . . . 233
9.33.1 Detailed Description . . . 233
9.34 VirusMonitorStatus::alarms Struct Reference . . . 234

9.35 PFIPMonitorStatus::alarms Struct Reference . . . 235
9.36 Ui::alertsDialog Class Reference . . . 236
9.37 alertsDialog Class Reference . . . 237
9.38 altaz_coordinate Class Reference . . . 238
9.38.1 Detailed Description . . . 238
9.39 Amoeba< Func > Class Template Reference . . . 239
9.39.1 Member Function Documentation . . . 239
9.39.1.1 amotry . . . 239
9.40 Amp Class Reference . . . 240
9.41 AnalysisRegion Struct Reference . . . 242
9.42 apcCmd Class Reference . . . 243
9.43 apcCmdConfig Class Reference . . . 244
9.43.1 Member Function Documentation . . . 244
9.43.1.1 print_summary . . . 244
9.43.1.2 set_options . . . 244
9.44 apcConfig Class Reference . . . 245
9.44.1 Member Function Documentation . . . 245
9.44.1.1 print_summary . . . 245
9.44.1.2 set_options . . . 245
9.45 APCData Class Reference . . . 246
9.46 APCException Class Reference . . . 247
9.47 apcPduSubSystem Class Reference . . . 248
9.48 apcServer Class Reference . . . 249
9.49 Ui::apcTab Class Reference . . . 251
9.50 apcTabUi Class Reference . . . 252
9.51 tcs_photometry::ApperturePhotResults Struct Reference . . . 254
9.51.1 Detailed Description . . . 254
9.52 apply_offset Class Reference . . . 255
9.52.1 Detailed Description . . . 255
9.53 Archon Class Reference . . . 256
9.53.1 Detailed Description . . . 256
9.54 ArchonHandler Class Reference . . . 257
9.55 array_dbr_int_t Class Reference . . . 258
9.56 array_dbr_long_t Class Reference . . . 259
9.57 ArrayType Class Reference . . . 260
9.58 tcsdb::attr_descriptor Class Reference . . . 261

9.59 pytcs::tcsdb::attr_descriptor Class Reference . . . 262
9.60 averager< Type > Class Template Reference . . . 263
9.61 AVT Struct Reference . . . 264
9.62 BeginRangeProbes Class Reference . . . 265
9.62.1 Detailed Description . . . 265
9.63 bestDistance Struct Reference . . . 266
9.64 ExposureInfo::BINNING Struct Reference . . . 267
9.65 Bitfield Class Reference . . . 268
9.66 BuildPupilThreadArgs Struct Reference . . . 269
9.67 BusyHandler Class Reference . . . 270
9.67.1 Member Function Documentation . . . 270
9.67.1.1 process . . . 270
9.68 Camera Class Reference . . . 271
9.69 AcqCamStatus::camera_status Struct Reference . . . 273
9.70 CameraAVT Class Reference . . . 274
9.71 CameraConfig Class Reference . . . 275
9.71.1 Member Function Documentation . . . 277
9.71.1.1 set_options . . . 277
9.72 Ui::cameraControlTab Class Reference . . . 278
9.73 cameraControlTabUi Class Reference . . . 279
9.74 CameraFLI::CameraData Struct Reference . . . 280
9.75 CameraFLI Class Reference . . . 281
9.76 cameraImageMarker Class Reference . . . 282
9.77 cameraImageScene Class Reference . . . 283
9.78 Ui::cameraImageTab Class Reference . . . 286
9.79 cameraImageTabUi Class Reference . . . 287
9.80 cameraImageTabUiThreadWorker Class Reference . . . 289
9.81 CameraInfo Class Reference . . . 290
9.82 Ui::cameraMetrologyControlTab Class Reference . . . 291
9.83 cameraMetrologyControlTabUi Class Reference . . . 292
9.84 Ui::cameraMetrologyDataTab Class Reference . . . 293
9.85 cameraMetrologyDataTabUi Class Reference . . . 294
9.86 CameraPGR Class Reference . . . 295
9.87 CameraSim Class Reference . . . 297
9.88 CAMRAConfig Class Reference . . . 298
9.88.1 Member Function Documentation . . . 298

9.88.1.1 print_summary . . . 298
9.88.1.2 set_options . . . 299
9.89 CAMRADriver Class Reference . . . 300
9.89.1 Detailed Description . . . 300
9.89.2 Member Function Documentation . . . 301
9.89.2.1 exposure . . . 301
9.89.2.2 populate_onewire_device_strings . . . 301
9.89.2.3 reset_tcs . . . 301
9.89.2.4 stash_temps . . . 301
9.90 CAMRAException Class Reference . . . 302
9.91 camra_handlers::CAMRAHandler Class Reference . . . 303
9.91.1 Detailed Description . . . 303
9.92 CAMRAHardware Class Reference . . . 304
9.92.1 Detailed Description . . . 304
9.93 cancel_trajectory Class Reference . . . 307
9.93.1 Detailed Description . . . 307
9.93.2 Member Function Documentation . . . 307
9.93.2.1 process . . . 307
9.94 CBUF< indexType, size, entryType > Class Template Reference . . . 308
9.95 geHeaters::Ccd Class Reference . . . 309
9.96 CCD Class Reference . . . 310
9.97 center Class Reference . . . 312
9.97.1 Detailed Description . . . 312
9.97.2 Member Function Documentation . . . 312
9.97.2.1 process . . . 312
9.98 CentroidInfo Class Reference . . . 313
9.99 check Class Reference . . . 314
9.99.1 Member Function Documentation . . . 314
9.99.1.1 process . . . 314
9.100 chunkHeader Class Reference . . . 315
9.101 cJSON Struct Reference . . . 316
9.102 cJSON_Hooks Struct Reference . . . 317
9.103 CleanerConfig Class Reference . . . 318
9.103.1 Member Function Documentation . . . 318
9.103.1.1 set_options . . . 318
9.104 clear_fiducial Class Reference . . . 319

9.104.1 Detailed Description . . . 319
9.104.2 Member Function Documentation . . . 319
9.104.2.1 process . . . 319
9.105clear_trajectories Class Reference . . . 320
9.105.1 Detailed Description . . . 320
9.105.2 Member Function Documentation . . . 320
9.105.2.1 process . . . 320
9.106clearError Class Reference . . . 321
9.106.1 Detailed Description . . . 321
9.107tracker_handlers::clearError Class Reference . . . 322
9.107.1 Member Function Documentation . . . 322
9.107.1.1 process . . . 322
9.108Client Class Reference . . . 323
9.108.1 Detailed Description . . . 323
9.109Close Class Reference . . . 324
9.109.1 Detailed Description . . . 324
9.110camra_handlers::close_shutter Class Reference . . . 325
9.110.1 Member Function Documentation . . . 325
9.110.1.1 process . . . 325
9.111close_shutter Class Reference . . . 326
9.111.1 Detailed Description . . . 326
9.112CloseShutter Class Reference . . . 327
9.112.1 Detailed Description . . . 327
9.113cmdConfig Class Reference . . . 328
9.113.1 Detailed Description . . . 328
9.113.2 Member Function Documentation . . . 328
9.113.2.1 print_summary . . . 328
9.113.2.2 set_options . . . 328
9.114Cntl Class Reference . . . 329
9.115coefs_t Struct Reference . . . 331
9.116Command Class Reference . . . 332
9.117camra_handlers::command_controller Class Reference . . . 333
9.117.1 Member Function Documentation . . . 333
9.117.1.1 process . . . 333
9.118command_controller Class Reference . . . 334
9.118.1 Detailed Description . . . 334

9.119tthread::condition_variable Class Reference . . . 335
9.119.1 Detailed Description . . . 335
9.119.2 Member Function Documentation . . . 335
9.119.2.1 notify_all . . . 335
9.119.2.2 notify_one . . . 336
9.119.2.3 wait . . . 336
9.119.2.4 wait_for . . . 336
9.120Config Class Reference . . . 337
9.120.1 Detailed Description . . . 337
9.121ConfigHandler Class Reference . . . 338
9.122tcs_photometry::ConfigureParams Struct Reference . . . 339
9.122.1 Detailed Description . . . 339
9.123wfs_photometry::ConfigureParams Struct Reference . . . 340
9.124Ui::configureSystemsDialog Class Reference . . . 341
9.125configureSystemsDialog Class Reference . . . 342
9.126configureTabsDialog Class Reference . . . 343
9.127Ui::configureTabsDialog Class Reference . . . 344
9.128confirmOffsetDialog Class Reference . . . 345
9.129Ui::confirmOffsetDialog Class Reference . . . 346
9.130consoleConfig Class Reference . . . 347
9.130.1 Member Function Documentation . . . 347
9.130.1.1 print_summary . . . 347
9.130.1.2 set_options . . . 347
9.131ControllerConfig Class Reference . . . 348
9.131.1 Member Function Documentation . . . 348
9.131.1.1 print_summary . . . 348
9.131.1.2 set_options . . . 348
9.132CoordinateTransformException Class Reference . . . 349
9.133correct_for_structure Class Reference . . . 350
9.133.1 Detailed Description . . . 350
9.133.2 Member Function Documentation . . . 350
9.133.2.1 process . . . 350
9.134Correction Class Reference . . . 351
9.134.1 Detailed Description . . . 351
9.135cubFit Class Reference . . . 352
9.136AcqCWFSStatus::cwfs_status Struct Reference . . . 353

9.137CWFSInfo Class Reference . . . 354
9.138CyclePipeline Class Reference . . . 355
9.138.1 Detailed Description . . . 355
9.139mainWindow::DashBoardAlerts Class Reference . . . 356
9.140Payload::Data Class Reference . . . 357
9.141dataChunk Class Reference . . . 358
9.142deactivate Class Reference . . . 359
9.142.1 Detailed Description . . . 359
9.142.2 Member Function Documentation . . . 359
9.142.2.1 process . . . 359
9.143deep_arg_t Struct Reference . . . 360
9.144delta_time Struct Reference . . . 362
9.145DeployACQCameraMirror Class Reference . . . 363
9.145.1 Detailed Description . . . 363
9.146DeployCWFSCamera Class Reference . . . 364
9.146.1 Detailed Description . . . 364
9.147DeployCWFSCameraMirror Class Reference . . . 365
9.147.1 Detailed Description . . . 365
9.148DeployPVCamera Class Reference . . . 366
9.148.1 Detailed Description . . . 366
9.149detachedTabWindow Class Reference . . . 367
9.150DeviceConfig Class Reference . . . 368
9.150.1 Member Function Documentation . . . 368
9.150.1.1 set_options . . . 368
9.151DevicePipelineEnableCCASDMIHandlerClass Class Reference . . . 369
9.151.1 Member Function Documentation . . . 369
9.151.1.1 process . . . 369
9.152DevicePipelineEnableSACDMIHandlerClass Class Reference . . . 370
9.152.1 Member Function Documentation . . . 370
9.152.1.1 process . . . 370
9.153DevicePipelineSetGROCHandlerClass Class Reference . . . 371
9.153.1 Member Function Documentation . . . 371
9.153.1.1 process . . . 371
9.154DevicePipelineStatusHandlerClass Class Reference . . . 372
9.154.1 Member Function Documentation . . . 372
9.154.1.1 process . . . 372

9.155camra_handlers::disable_pressure_gauges Class Reference . . . 373
9.155.1 Member Function Documentation . . . 373
9.155.1.1 process . . . 373
9.156disable_pressure_gauges Class Reference . . . 374
9.156.1 Detailed Description . . . 374
9.157disable_rho Class Reference . . . 375
9.157.1 Detailed Description . . . 375
9.157.2 Member Function Documentation . . . 375
9.157.2.1 process . . . 375
9.158camra_handlers::disable_shutter Class Reference . . . 376
9.158.1 Member Function Documentation . . . 376
9.158.1.1 process . . . 376
9.159disable_shutter Class Reference . . . 377
9.159.1 Detailed Description . . . 377
9.160DisableProbes Class Reference . . . 378
9.160.1 Detailed Description . . . 378
9.161DMI_get_status Class Reference . . . 379
9.161.1 Detailed Description . . . 379
9.161.2 Member Function Documentation . . . 379
9.161.2.1 process . . . 379
9.162pasDevicePipelineSubSystem::DMIData Struct Reference . . . 380
9.163DMIData Class Reference . . . 381
9.164DMIHandler Class Reference . . . 382
9.165DMILoop Class Reference . . . 383
9.166liborchestrate::DmiSubsystem Class Reference . . . 384
9.167Ui::dmiTab Class Reference . . . 385
9.168dmiTabUi Class Reference . . . 386
9.169Doh Class Reference . . . 387
9.170dome_Abort Class Reference . . . 388
9.170.1 Detailed Description . . . 388
9.171dome_GetPosition Class Reference . . . 389
9.172dome_Home Class Reference . . . 390
9.172.1 Detailed Description . . . 390
9.173dome_IssuePMACCmd Class Reference . . . 391
9.173.1 Detailed Description . . . 391
9.174dome_MoveTo Class Reference . . . 392

9.174.1 Detailed Description . . . 392
9.175dome_MoveToCCAS Class Reference . . . 393
9.176dome_OnLine Class Reference . . . 394
9.176.1 Detailed Description . . . 394
9.177dome_Reset Class Reference . . . 395
9.177.1 Detailed Description . . . 395
9.178dome_SetDirection Class Reference . . . 396
9.178.1 Detailed Description . . . 396
9.179dome_SetPower Class Reference . . . 397
9.179.1 Detailed Description . . . 397
9.180dome_SetVelocity Class Reference . . . 398
9.180.1 Detailed Description . . . 398
9.181dome_Status Class Reference . . . 399
9.181.1 Detailed Description . . . 399
9.182dome_Stop Class Reference . . . 400
9.182.1 Detailed Description . . . 400
9.183dome_Stow Class Reference . . . 401
9.184dome_Test Class Reference . . . 402
9.184.1 Detailed Description . . . 402
9.185DomeClient Class Reference . . . 403
9.186DomeConfig Class Reference . . . 404
9.186.1 Member Function Documentation . . . 404
9.186.1.1 set_options . . . 404
9.187DomeStatus Class Reference . . . 405
9.188DomeStatusHandlerClass Class Reference . . . 406
9.188.1 Member Function Documentation . . . 406
9.188.1.1 process . . . 406
9.189Ui::domeTab Class Reference . . . 407
9.190domeTabUi Class Reference . . . 408
9.191droppablePlainTextEdit Class Reference . . . 409
9.192tthread::chrono::duration< _Rep, _Period > Class Template Reference . . . 410
9.192.1 Detailed Description . . . 410
9.193DynamicConfig Struct Reference . . . 411
9.194editPreferencesDialog Class Reference . . . 412
9.195Ui::editPreferencesDialog Class Reference . . . 413
9.196enable_pressure_gauges Class Reference . . . 414

9.196.1 Detailed Description . . . 414
9.197 camra_handlers::enable_pressure_gauges Class Reference . . . 415
9.197.1 Member Function Documentation . . . 415
9.197.1.1 process . . . 415
9.198 enable_rho Class Reference . . . 416
9.198.1 Detailed Description . . . 416
9.198.2 Member Function Documentation . . . 416
9.198.2.1 process . . . 416
9.199 camra_handlers::enable_shutter Class Reference . . . 417
9.199.1 Member Function Documentation . . . 417
9.199.1.1 process . . . 417
9.200 enable_shutter Class Reference . . . 418
9.200.1 Detailed Description . . . 418
9.201 EnableCCASDMI Class Reference . . . 419
9.202 EnableProbes Class Reference . . . 420
9.202.1 Detailed Description . . . 420
9.203 EnableSACDMI Class Reference . . . 421
9.204 EndRangeProbes Class Reference . . . 422
9.204.1 Detailed Description . . . 422
9.205 Enum Class Reference . . . 423
9.206 AcqENWAStatus::enwa_status Struct Reference . . . 424
9.207 ENWAInfo Class Reference . . . 425
9.208 Ephemeris Struct Reference . . . 426
9.208.1 Detailed Description . . . 426
9.209 EphList Struct Reference . . . 427
9.210 EphRec Struct Reference . . . 428
9.211 EPICSComm Class Reference . . . 429
9.211.1 Detailed Description . . . 429
9.211.2 Constructor & Destructor Documentation . . . 430
9.211.2.1 EPICSComm . . . 430
9.211.3 Member Function Documentation . . . 430
9.211.3.1 Connect . . . 430
9.211.3.2 GetString . . . 430
9.211.3.3 getValues . . . 430
9.211.3.4 SetInt . . . 430
9.211.3.5 setValues . . . 431

9.212EPICSException Class Reference . . . 432
9.213EPICSHandle Class Reference . . . 433
9.214EPICSRelayConfig Class Reference . . . 434
9.214.1 Member Function Documentation . . . 434
9.214.1.1 print_summary . . . 434
9.214.1.2 set_options . . . 434
9.215EPICSRelayDriver Class Reference . . . 435
9.216eq_coords Class Reference . . . 436
9.217Event Class Reference . . . 437
9.217.1 Detailed Description . . . 437
9.217.2 Member Function Documentation . . . 437
9.217.2.1 operator<< . . . 437
9.217.2.2 operator<< . . . 437
9.217.2.3 variable . . . 438
9.218replay::EventBuffer Class Reference . . . 439
9.219eventCmd Class Reference . . . 440
9.220EventHandler Class Reference . . . 441
9.220.1 Detailed Description . . . 441
9.220.2 Member Function Documentation . . . 441
9.220.2.1 add_system . . . 441
9.220.2.2 set_default . . . 441
9.220.2.3 set_global_defaults . . . 442
9.220.2.4 value . . . 442
9.220.2.5 wait . . . 442
9.221EventHeader Class Reference . . . 443
9.222EventList Class Reference . . . 444
9.223Ui::eventMonitorTab Class Reference . . . 445
9.224eventMonitorTabUi Class Reference . . . 446
9.225eventMonitorTabUiThreadWorker Class Reference . . . 447
9.226eventsConfig Class Reference . . . 448
9.226.1 Member Function Documentation . . . 448
9.226.1.1 print_summary . . . 448
9.226.1.2 set_options . . . 448
9.227EventSink Class Reference . . . 449
9.227.1 Detailed Description . . . 449
9.227.2 Constructor & Destructor Documentation . . . 449

9.227.2.1 EventSink . . . 449
9.227.3 Member Function Documentation . . . 449
9.227.3.1 add_system . . . 449
9.227.3.2 disable . . . 449
9.227.3.3 next_event . . . 449
9.227.3.4 set_filter . . . 450
9.228 EventSource Class Reference . . . 451
9.228.1 Detailed Description . . . 451
9.228.2 Member Function Documentation . . . 451
9.228.2.1 name . . . 451
9.229 EventSourceRoot Class Reference . . . 452
9.229.1 Detailed Description . . . 452
9.230 EventSourceSocket Class Reference . . . 453
9.230.1 Detailed Description . . . 453
9.231 ExportData Class Reference . . . 454
9.232 camra_handlers::expose Class Reference . . . 455
9.232.1 Member Function Documentation . . . 455
9.232.1.1 process . . . 455
9.233 expose Class Reference . . . 456
9.233.1 Detailed Description . . . 456
9.234 Exposure Class Reference . . . 457
9.234.1 Detailed Description . . . 457
9.235 ExposureDescriptor Struct Reference . . . 458
9.235.1 Detailed Description . . . 458
9.236 ExposureInfo Struct Reference . . . 460
9.237 tthread::fast_mutex Class Reference . . . 462
9.237.1 Detailed Description . . . 462
9.237.2 Member Function Documentation . . . 462
9.237.2.1 lock . . . 462
9.237.2.2 try_lock . . . 462
9.237.2.3 unlock . . . 463
9.238 FCU_Lamp Class Reference . . . 464
9.239 FCU_LDLS Class Reference . . . 465
9.240 FCU_Shutter Class Reference . . . 466
9.241 FCU_Slider Class Reference . . . 467
9.242 FCU_State Class Reference . . . 468

9.243 FCUStatus::fcu_status Struct Reference . . . 470
9.244 FCU_SubSystem Class Reference . . . 472
9.244.1 Member Function Documentation . . . 473
9.244.1.1 BroadcastStatusEvents . . . 473
9.244.1.2 EnableLamps . . . 473
9.244.1.3 ExposureSetup . . . 473
9.244.1.4 MoveOutputLightGuide . . . 473
9.244.1.5 MoveSelectorPosition . . . 473
9.244.1.6 ReplyCurrentExposure . . . 473
9.244.1.7 ReplyLastExposure . . . 474
9.244.1.8 SetValue . . . 474
9.245 FCUCancelExposure Class Reference . . . 475
9.245.1 Detailed Description . . . 475
9.246 FCUCommException Class Reference . . . 476
9.247 FCUCommunication Class Reference . . . 477
9.248 Ui::fcuControlTab Class Reference . . . 478
9.249 fcuControlTabUi Class Reference . . . 479
9.250 FCUDeployHead Class Reference . . . 480
9.250.1 Detailed Description . . . 480
9.251 FCUDisableAllLights Class Reference . . . 481
9.251.1 Detailed Description . . . 481
9.252 FCUEnableLamps Class Reference . . . 482
9.252.1 Detailed Description . . . 482
9.253 FCUEnableLightSources Class Reference . . . 483
9.253.1 Detailed Description . . . 483
9.254 FCUException Class Reference . . . 484
9.255 FCUExposeLDLS Class Reference . . . 485
9.255.1 Detailed Description . . . 485
9.256 FCUExposeSliderA Class Reference . . . 486
9.256.1 Detailed Description . . . 486
9.257 FCUExposeSliderB Class Reference . . . 487
9.257.1 Detailed Description . . . 487
9.258 FCUGetCurrentExposureInfo Class Reference . . . 488
9.258.1 Detailed Description . . . 488
9.259 FCUGetLastExposureInfo Class Reference . . . 489
9.259.1 Detailed Description . . . 489

9.260FCUHandler Class Reference 490
9.260.1 Constructor & Destructor Documentation 490
9.260.1.1 FCUHandler 490
9.260.2 Member Function Documentation 490
9.260.2.1 FCUExposeSlider 490
9.260.2.2 parseSlider 491
9.260.2.3 parseSliderPos 491
9.261FCUInitLightSources Class Reference 492
9.261.1 Detailed Description 492
9.262FCUMoveLightBank Class Reference 493
9.262.1 Detailed Description 493
9.263FCUMoveLightGuide Class Reference 494
9.263.1 Detailed Description 494
9.264FCUMoveSlider Class Reference 495
9.264.1 Detailed Description 495
9.265FCUMutexException Class Reference 496
9.266FCUPrintRegister Class Reference 497
9.266.1 Detailed Description 497
9.267FCURetractHead Class Reference 498
9.267.1 Detailed Description 498
9.268FCUSetLampWarmUp Class Reference 499
9.268.1 Detailed Description 499
9.269FCUSetLDLSWarmUp Class Reference 500
9.269.1 Detailed Description 500
9.270FCUShutDown Class Reference 501
9.270.1 Detailed Description 501
9.271FCUStatus Class Reference 502
9.272Ui::fcuStatusTab Class Reference 503
9.273fcuStatusTabUi Class Reference 504
9.274FCUWriteRegister Class Reference 505
9.274.1 Detailed Description 505
9.275fiducialDialog Class Reference 506
9.276Ui::fiducialDialog Class Reference 507
9.277pasImagePipelineSubSystem::TCSData::fiducialObject Struct Reference 508
9.278ExposureInfo::FILTER_INFO Struct Reference 509
9.279AcqCamStatus::filter_status Struct Reference 510

9.280 Filters Class Reference . . . 511
9.281 FitData Class Reference . . . 512
9.282 tcs_photometry::FitPars Struct Reference . . . 513
9.282.1 Detailed Description . . . 513
9.283 pasImagePipelineSubSystem::FITSMapEntry Struct Reference . . . 514
9.284 FitsOutWrapper Class Reference . . . 515
9.285 fixed_point::FixedPoint< T, N > Class Template Reference . . . 516
9.285.1 Detailed Description . . . 516
9.285.2 Constructor & Destructor Documentation . . . 517
9.285.2.1 FixedPoint . . . 517
9.285.2.2 FixedPoint . . . 517
9.286 PFIPStatusBase::flags Struct Reference . . . 518
9.287 flagValues Class Reference . . . 519
9.288 focalPlaneShutterScene Class Reference . . . 520
9.289 Ui::focalPlaneShutterTab Class Reference . . . 521
9.290 focalPlaneShutterTabUi Class Reference . . . 522
9.291 Foo Class Reference . . . 523
9.292 FooBar Class Reference . . . 524
9.293 force_state_event Class Reference . . . 525
9.293.1 Detailed Description . . . 525
9.293.2 Member Function Documentation . . . 525
9.293.2.1 process . . . 525
9.294 ForceStatusEvent Class Reference . . . 526
9.294.1 Detailed Description . . . 526
9.295 ForceStatusEventsHandlerClass Class Reference . . . 527
9.295.1 Member Function Documentation . . . 527
9.295.1.1 process . . . 527
9.295.1.2 process . . . 527
9.296 formatChunk Class Reference . . . 528
9.297 fptime Class Reference . . . 529
9.297.1 Detailed Description . . . 529
9.297.2 Constructor & Destructor Documentation . . . 529
9.297.2.1 fptime . . . 529
9.297.2.2 fptime . . . 529
9.297.2.3 fptime . . . 529
9.297.2.4 fptime . . . 530

9.297.2.5 fptime . . . 530
9.297.2.6 fptime . . . 530
9.297.3 Member Function Documentation . . . 530
9.297.3.1 as_utc . . . 530
9.297.3.2 from_decimal_equinox . . . 530
9.297.3.3 now . . . 530
9.298 fw102b::FW102B Class Reference . . . 531
9.299 GenerateExceptionHandler Class Reference . . . 532
9.299.1 Member Function Documentation . . . 532
9.299.1.1 process . . . 532
9.300 get Class Reference . . . 533
9.300.1 Member Function Documentation . . . 533
9.300.1.1 process . . . 533
9.300.1.2 process . . . 533
9.301 get_analysis_region Class Reference . . . 534
9.301.1 Detailed Description . . . 534
9.301.2 Member Function Documentation . . . 534
9.301.2.1 process . . . 534
9.302 camra_handlers::get_ccd_temp Class Reference . . . 535
9.302.1 Detailed Description . . . 535
9.302.2 Member Function Documentation . . . 535
9.302.2.1 process . . . 535
9.303 get_ccd_temp Class Reference . . . 536
9.303.1 Detailed Description . . . 536
9.304 camra_handlers::get_clock_voltages Class Reference . . . 537
9.304.1 Member Function Documentation . . . 537
9.304.1.1 process . . . 537
9.305 get_clock_voltages Class Reference . . . 538
9.305.1 Detailed Description . . . 538
9.306 get_correction Class Reference . . . 539
9.306.1 Detailed Description . . . 539
9.306.2 Member Function Documentation . . . 539
9.306.2.1 process . . . 539
9.307 camra_handlers::get_cryo_pressure Class Reference . . . 540
9.307.1 Member Function Documentation . . . 540
9.307.1.1 process . . . 540

9.308 get_cryo_pressure Class Reference . . . 541
9.308.1 Detailed Description . . . 541
9.309 camra_handlers::get_cryo_temp Class Reference . . . 542
9.309.1 Member Function Documentation . . . 542
9.309.1.1 process . . . 542
9.310 get_cryo_temp Class Reference . . . 543
9.310.1 Detailed Description . . . 543
9.311 get_current_local_sidereal_time Class Reference . . . 544
9.311.1 Detailed Description . . . 544
9.311.2 Member Function Documentation . . . 544
9.311.2.1 process . . . 544
9.312 get_current_pointing Class Reference . . . 545
9.312.1 Detailed Description . . . 545
9.312.2 Member Function Documentation . . . 545
9.312.2.1 process . . . 545
9.313 get_current_trajectory_info Class Reference . . . 546
9.313.1 Detailed Description . . . 546
9.313.2 Member Function Documentation . . . 546
9.313.2.1 process . . . 546
9.314 camra_handlers::get_exposure_info Class Reference . . . 547
9.314.1 Member Function Documentation . . . 547
9.314.1.1 process . . . 547
9.315 get_exposure_info Class Reference . . . 548
9.315.1 Detailed Description . . . 548
9.316 get_fiducial Class Reference . . . 549
9.316.1 Detailed Description . . . 549
9.316.2 Member Function Documentation . . . 549
9.316.2.1 process . . . 549
9.317 get_focus_gain Class Reference . . . 550
9.317.1 Detailed Description . . . 550
9.317.2 Member Function Documentation . . . 550
9.317.2.1 process . . . 550
9.318 get_focus_offset Class Reference . . . 551
9.318.1 Detailed Description . . . 551
9.318.2 Member Function Documentation . . . 551
9.318.2.1 process . . . 551

9.319 get_focus_window_length Class Reference . . . 552
9.319.1 Detailed Description . . . 552
9.319.2 Member Function Documentation . . . 552
9.319.2.1 process . . . 552
9.320 get_hardware_status Class Reference . . . 553
9.320.1 Detailed Description . . . 553
9.321 camra_handlers::get_hardware_status Class Reference . . . 554
9.321.1 Member Function Documentation . . . 554
9.321.1.1 process . . . 554
9.322 get_heater_voltage Class Reference . . . 555
9.322.1 Detailed Description . . . 555
9.323 camra_handlers::get_heater_voltage Class Reference . . . 556
9.323.1 Member Function Documentation . . . 556
9.323.1.1 process . . . 556
9.324 get_itf_offset Class Reference . . . 557
9.324.1 Detailed Description . . . 557
9.324.2 Member Function Documentation . . . 557
9.324.2.1 process . . . 557
9.325 get_metadata Class Reference . . . 558
9.325.1 Detailed Description . . . 558
9.325.2 Member Function Documentation . . . 558
9.325.2.1 process . . . 558
9.326 get_next_trajectory_info Class Reference . . . 559
9.326.1 Detailed Description . . . 559
9.326.2 Member Function Documentation . . . 559
9.326.2.1 process . . . 559
9.327 get_radec_offset Class Reference . . . 560
9.327.1 Detailed Description . . . 560
9.327.2 Member Function Documentation . . . 560
9.327.2.1 process . . . 560
9.328 get_sky_offset Class Reference . . . 561
9.328.1 Detailed Description . . . 561
9.328.2 Member Function Documentation . . . 561
9.328.2.1 process . . . 561
9.329 get_status Class Reference . . . 562
9.329.1 Detailed Description . . . 562

9.329.2 Member Function Documentation . . . 562
9.329.2.1 process . . . 562
9.330 get_tiptilt_gain Class Reference . . . 563
9.330.1 Detailed Description . . . 563
9.330.2 Member Function Documentation . . . 563
9.330.2.1 process . . . 563
9.331 get_tiptilt_offset Class Reference . . . 564
9.331.1 Detailed Description . . . 564
9.331.2 Member Function Documentation . . . 564
9.331.2.1 process . . . 564
9.332 get_tiptilt_window_length Class Reference . . . 565
9.332.1 Detailed Description . . . 565
9.332.2 Member Function Documentation . . . 565
9.332.2.1 process . . . 565
9.333 get_tuning< T > Class Template Reference . . . 566
9.333.1 Detailed Description . . . 566
9.333.2 Member Function Documentation . . . 566
9.333.2.1 process . . . 566
9.334 GetACQSystemInfo Class Reference . . . 567
9.334.1 Detailed Description . . . 567
9.335 GetAnalysisRegion Class Reference . . . 568
9.336 GetExposureTime Class Reference . . . 569
9.337 GetGain Class Reference . . . 570
9.338 GetLightSourcePowerLevel Class Reference . . . 571
9.339 GetOverScan Class Reference . . . 572
9.340 GetPhotPhotometryOptions Class Reference . . . 573
9.341 GetPosition Class Reference . . . 574
9.341.1 Detailed Description . . . 574
9.342 GetProbeInfo Class Reference . . . 575
9.342.1 Detailed Description . . . 575
9.343 GetProbePosition Class Reference . . . 576
9.343.1 Detailed Description . . . 576
9.344 GetRegionOfInterest Class Reference . . . 577
9.345 GetTemperatures Class Reference . . . 578
9.345.1 Detailed Description . . . 578
9.346 GetWeatherData Class Reference . . . 579

9.346.1 Detailed Description . . . 579
9.347 GetWFSPhotometryOptions Class Reference . . . 580
9.348 globals Struct Reference . . . 581
9.349 GLTrackerConfig Class Reference . . . 582
9.350 GLTrackerDriver Class Reference . . . 583
9.350.1 Member Function Documentation . . . 584
9.350.1.1 drawHexagon3D . . . 584
9.350.1.2 drawPrimaryMirror . . . 584
9.351 go_next Class Reference . . . 585
9.351.1 Detailed Description . . . 585
9.351.2 Member Function Documentation . . . 585
9.351.2.1 process . . . 585
9.352 GoNextFlags Class Reference . . . 586
9.352.1 Detailed Description . . . 586
9.353 GroupConfig Class Reference . . . 587
9.353.1 Member Function Documentation . . . 587
9.353.1.1 print_summary . . . 587
9.353.1.2 set_options . . . 587
9.354 guiConfig Class Reference . . . 588
9.354.1 Member Function Documentation . . . 589
9.354.1.1 print_summary . . . 589
9.354.1.2 print_summary . . . 589
9.354.1.3 set_options . . . 589
9.354.1.4 set_options . . . 589
9.355 guideBoxDialog Class Reference . . . 591
9.356 Ui::guideBoxDialog Class Reference . . . 592
9.357 GuideLoop Class Reference . . . 593
9.358 tcs_photometry::GuideRect Struct Reference . . . 594
9.358.1 Detailed Description . . . 594
9.359 GuiderHandler Class Reference . . . 595
9.360 tcs_photometry::GuiderMetrologyData Struct Reference . . . 596
9.361 hetdex-dither::GuiderRelay Class Reference . . . 597
9.361.1 Detailed Description . . . 597
9.361.2 Member Function Documentation . . . 597
9.361.2.1 __getattr__ . . . 597
9.362 GUIException Class Reference . . . 598

9.363guiPreferences Class Reference . . . 599
9.364guiSound Class Reference . . . 600
9.365liborchestrate::GuiSubsystem Class Reference . . . 601
9.366handler_t Struct Reference . . . 602
9.367HardwareElement Class Reference . . . 603
9.368headerChunk Class Reference . . . 604
9.369HelloFunctor Struct Reference . . . 605
9.370HelpHandler Class Reference . . . 606
9.370.1 Detailed Description . . . 606
9.370.2 Member Function Documentation . . . 606
9.370.2.1 process . . . 606
9.371hetdex-dither::HetdexDitherError Class Reference . . . 607
9.371.1 Detailed Description . . . 607
9.372HETEarthOrientation Class Reference . . . 608
9.372.1 Detailed Description . . . 608
9.372.2 Member Function Documentation . . . 608
9.372.2.1 configure . . . 608
9.372.2.2 lst . . . 608
9.372.2.3 lst_str . . . 608
9.372.2.4 pmx . . . 608
9.373HETEarthOrientationException Class Reference . . . 609
9.374HETillumination Class Reference . . . 610
9.375hex_fit Class Reference . . . 612
9.376Home Class Reference . . . 613
9.376.1 Detailed Description . . . 613
9.377HomeProbes Class Reference . . . 614
9.377.1 Detailed Description . . . 614
9.378HomeShutter Class Reference . . . 615
9.378.1 Detailed Description . . . 615
9.379HotPixel_t Class Reference . . . 616
9.380ICSConfig Class Reference . . . 617
9.380.1 Member Function Documentation . . . 617
9.380.1.1 name . . . 617
9.381ICSDaemon< S > Class Template Reference . . . 618
9.382ICSQuery Class Reference . . . 619
9.383tthread::thread::id Class Reference . . . 620

9.383.1 Detailed Description . . . 620
9.383.2 Constructor & Destructor Documentation . . . 620
9.383.2.1 id . . . 620
9.384 IglooData Class Reference . . . 621
9.385 IglooRegAlertList Class Reference . . . 622
9.385.1 Detailed Description . . . 622
9.385.2 Member Function Documentation . . . 622
9.385.2.1 get_messages . . . 622
9.385.2.2 initialize . . . 622
9.386 IglooSensors Class Reference . . . 623
9.386.1 Detailed Description . . . 623
9.387 IglooSensorsConfig Class Reference . . . 624
9.387.1 Member Function Documentation . . . 624
9.387.1.1 print_summary . . . 624
9.387.1.2 set_options . . . 624
9.388 IglooSensorsConnection Class Reference . . . 625
9.389 IglooSensorsException Class Reference . . . 626
9.390 IglooSensorsHandler Class Reference . . . 627
9.391 IglooStatusField Struct Reference . . . 628
9.391.1 Detailed Description . . . 628
9.392 IglooStatusFieldList Class Reference . . . 629
9.392.1 Detailed Description . . . 629
9.393 iglooStatusFieldWidgetSet Class Reference . . . 630
9.393.1 Constructor & Destructor Documentation . . . 630
9.393.1.1 iglooStatusFieldWidgetSet . . . 630
9.394 Ui::iglooTab Class Reference . . . 631
9.395 iglooTabUi Class Reference . . . 632
9.396 ExposureInfo::Image Struct Reference . . . 633
9.397 Image Class Reference . . . 634
9.398 ImageCleaner Class Reference . . . 635
9.399 ImageData Class Reference . . . 636
9.400 pasImagePipelineSubSystem::ImageHandlerData Struct Reference . . . 637
9.401 ImageMask Class Reference . . . 638
9.402 ImagePipelineGetExposureTimeHandlerClass Class Reference . . . 639
9.402.1 Member Function Documentation . . . 639
9.402.1.1 process . . . 639

9.403 ImagePipelineGetGainHandlerClass Class Reference . . . 640
9.403.1 Member Function Documentation . . . 640
9.403.1.1 process . . . 640
9.404 ImagePipelineGetOverScanHandlerClass Class Reference . . . 641
9.404.1 Member Function Documentation . . . 641
9.404.1.1 process . . . 641
9.405 ImagePipelineGetPhotPhotometryOptionsHandlerClass Class Reference . . . 642
9.405.1 Member Function Documentation . . . 642
9.405.1.1 process . . . 642
9.406 ImagePipelineGetRegionOfInterestHandlerClass Class Reference . . . 643
9.406.1 Member Function Documentation . . . 643
9.406.1.1 process . . . 643
9.407 ImagePipelineGetWFSPhotometryOptionsHandlerClass Class Reference . . . 644
9.407.1 Member Function Documentation . . . 644
9.407.1.1 process . . . 644
9.408 ImagePipelineSetCleanImagesHandlerClass Class Reference . . . 645
9.408.1 Member Function Documentation . . . 645
9.408.1.1 process . . . 645
9.409 ImagePipelineSetCleaningBiasTypeHandlerClass Class Reference . . . 646
9.409.1 Member Function Documentation . . . 646
9.409.1.1 process . . . 646
9.410 ImagePipelineSetFilterExposureTimeFactorHandlerClass Class Reference . . . 647
9.410.1 Member Function Documentation . . . 647
9.410.1.1 process . . . 647
9.411 ImagePipelineSetFilterSequenceHandlerClass Class Reference . . . 648
9.411.1 Member Function Documentation . . . 648
9.411.1.1 process . . . 648
9.412 ImagePipelineSetFilterZeroPointHandlerClass Class Reference . . . 649
9.412.1 Member Function Documentation . . . 649
9.412.1.1 process . . . 649
9.413 ImagePipelineSetObjectAndMagnitudesHandlerClass Class Reference . . . 650
9.413.1 Member Function Documentation . . . 650
9.413.1.1 process . . . 650
9.414 ImagePipelineSetOverScanHandlerClass Class Reference . . . 651
9.414.1 Member Function Documentation . . . 651
9.414.1.1 process . . . 651

9.415 ImagePipelineSetPhotometryOptionsHandlerClass Class Reference 652
9.415.1 Member Function Documentation 652
9.415.1.1 process 652
9.416 ImagePipelineSetPhotPhotometryOptionsHandlerClass Class Reference 653
9.416.1 Member Function Documentation 653
9.416.1.1 process 653
9.417 ImagePipelineSetProcessingTypeHandlerClass Class Reference 654
9.417.1 Member Function Documentation 654
9.417.1.1 process 654
9.418 ImagePipelineSetWFSPhotometryOptionsHandlerClass Class Reference 655
9.418.1 Member Function Documentation 655
9.418.1.1 process 655
9.419 ImagePipelineStartSendingImagesHandlerClass Class Reference 656
9.419.1 Member Function Documentation 656
9.419.1.1 process 656
9.420 ImagePipelineStatus Class Reference 657
9.421 ImageProcessor Class Reference 658
9.422 imageRect< Type > Class Template Reference 659
9.423 index_time Class Reference 660
9.424 init_t Struct Reference 661
9.425 initTMCS Class Reference 662
9.425.1 Detailed Description 662
9.426 tracker_handlers::initTMCS Class Reference 663
9.426.1 Member Function Documentation 663
9.426.1.1 process 663
9.427 inject_metrology Class Reference 664
9.427.1 Detailed Description 664
9.427.2 Member Function Documentation 664
9.427.2.1 process 664
9.428 Instrument Class Reference 665
9.428.1 Member Function Documentation 665
9.428.1.1 name 665
9.429 InterruptHandler Class Reference 666
9.429.1 Detailed Description 666
9.429.2 Member Function Documentation 666
9.429.2.1 process 666

9.430fixed_point::fixed_internal::ioflags Class Reference 667
9.430.1 Detailed Description 667
9.431IssuePMACCmd Class Reference 668
9.431.1 Detailed Description 668
9.432itf_coordinate Struct Reference 669
9.432.1 Detailed Description 669
9.433itf_tsf Class Reference 670
9.433.1 Detailed Description 670
9.434julian_day Class Reference 671
9.435tcsdb::key_descriptor Class Reference 672
9.436pytcs::tcsdb::key_descriptor Class Reference 673
9.437legacy_ForceStatusEvents Class Reference 674
9.438legacyConfig Class Reference 675
9.438.1 Member Function Documentation 675
9.438.1.1 print_summary 675
9.438.1.2 set_options 675
9.439liborchestrate::LegacyDomeSimulator Class Reference 676
9.440legacyDomeSubSystem Class Reference 677
9.440.1 Detailed Description 677
9.441LegacyException Class Reference 680
9.442legacyServer Class Reference 681
9.443liborchestrate::LegacyShutterSimulator Class Reference 682
9.444legacyShutterSubSystem Class Reference 683
9.444.1 Detailed Description 683
9.445liborchestrate::LegacyStructureSimulator Class Reference 685
9.446legacyStructureSubSystem Class Reference 686
9.446.1 Detailed Description 686
9.447liborchestrate::LegacySubsystem Class Reference 689
9.448Ui::legacyTab Class Reference 690
9.449legacyTabUi Class Reference 691
9.450legacyWeatherSubSystem Class Reference 692
9.450.1 Detailed Description 692
9.451LightSource Class Reference 695
9.452LightSourceConfig Class Reference 696
9.452.1 Member Function Documentation 696
9.452.1.1 set_options 696

9.453 linFit Class Reference . . . 697
9.453.1 Detailed Description . . . 697
9.454 load_trajectory Class Reference . . . 698
9.454.1 Detailed Description . . . 698
9.454.2 Member Function Documentation . . . 698
9.454.2.1 process . . . 698
9.455 loadedUi Class Reference . . . 699
9.456 loadMarkersDialog Class Reference . . . 700
9.457 Ui::loadMarkersDialog Class Reference . . . 701
9.458 tthread::lock_guard< T > Class Template Reference . . . 702
9.458.1 Detailed Description . . . 702
9.459 LogClientException Class Reference . . . 703
9.460 Logger Class Reference . . . 704
9.460.1 Detailed Description . . . 704
9.460.2 Member Function Documentation . . . 705
9.460.2.1 fmtConsole . . . 705
9.460.2.2 makeFileLoc . . . 705
9.460.2.3 SendExternalLogMessage . . . 705
9.460.2.4 SendLogMessage . . . 705
9.461 LoggerDB Class Reference . . . 707
9.462 LogRelayConfig Class Reference . . . 708
9.463 LogRelayDriver Class Reference . . . 709
9.463.1 Member Function Documentation . . . 709
9.463.1.1 process . . . 709
9.464 LogServerException Class Reference . . . 710
9.465 HETillumination::M1motion Struct Reference . . . 711
9.466 HETillumination::M4motion Struct Reference . . . 712
9.467 m7302::M7302 Class Reference . . . 713
9.468 mainWindow Class Reference . . . 714
9.469 Ui::mainWindow Class Reference . . . 717
9.470 TCSConfig::mapEntry Struct Reference . . . 718
9.471 MAXONStatusInfo Class Reference . . . 719
9.472 mbintf_bitfield_t Struct Reference . . . 720
9.472.1 Detailed Description . . . 720
9.472.2 Member Data Documentation . . . 720
9.472.2.1 alrtlvl . . . 720

9.472.2.2 bit . . . 720
9.472.2.3 desc . . . 720
9.472.2.4 label . . . 720
9.472.2.5 normal . . . 720
9.473 mbintf_hldreg_bitfield_t Struct Reference . . . 721
9.473.1 Detailed Description . . . 721
9.473.2 Member Data Documentation . . . 721
9.473.2.1 bitfields . . . 721
9.473.2.2 bitfields_ct . . . 721
9.473.2.3 label . . . 721
9.473.2.4 mreg_id . . . 721
9.473.2.5 reg_id . . . 721
9.474 mbintf_inpreg_analog_t Struct Reference . . . 722
9.474.1 Detailed Description . . . 722
9.474.2 Member Data Documentation . . . 722
9.474.2.1 alrtlvl . . . 722
9.474.2.2 desc . . . 722
9.474.2.3 label . . . 722
9.474.2.4 limit_lower . . . 722
9.474.2.5 limit_upper . . . 722
9.474.2.6 reg_id . . . 722
9.474.2.7 regval_div . . . 722
9.474.2.8 regval_mult . . . 723
9.475 mbintf_inpreg_bitfield_t Struct Reference . . . 724
9.475.1 Detailed Description . . . 724
9.475.2 Member Data Documentation . . . 724
9.475.2.1 bitfields . . . 724
9.475.2.2 bitfields_ct . . . 724
9.475.2.3 label . . . 724
9.475.2.4 reg_id . . . 724
9.476 mbreg_ptr_t Union Reference . . . 725
9.476.1 Detailed Description . . . 725
9.477 mbreg_t Struct Reference . . . 726
9.477.1 Detailed Description . . . 726
9.477.2 Member Function Documentation . . . 726
9.477.2.1 get_name . . . 726

9.477.3 Member Data Documentation . . . 726
9.477.3.1 m_bit_id . . . 726
9.477.3.2 m_is_hld . . . 726
9.477.3.3 m_regptr . . . 726
9.478 mbregid_t Union Reference . . . 728
9.478.1 Detailed Description . . . 728
9.479 MBRegVal Class Reference . . . 729
9.479.1 Detailed Description . . . 729
9.480 md5::MD5 Class Reference . . . 730
9.481 Message Class Reference . . . 731
9.481.1 Detailed Description . . . 731
9.481.2 Member Function Documentation . . . 732
9.481.2.1 done . . . 732
9.481.2.2 get_origin . . . 732
9.481.2.3 get_receive_time . . . 732
9.481.2.4 id . . . 732
9.481.2.5 interrupt . . . 732
9.481.2.6 interrupt_mark . . . 732
9.481.2.7 is_done . . . 732
9.481.2.8 is_error . . . 732
9.481.2.9 respond . . . 733
9.482 MessageHandler Class Reference . . . 734
9.482.1 Detailed Description . . . 734
9.482.2 Member Function Documentation . . . 734
9.482.2.1 process . . . 734
9.483 MessageHandlerDescriptor Class Reference . . . 736
9.483.1 Detailed Description . . . 736
9.484 MessageHeader Class Reference . . . 737
9.485 MessagingException Class Reference . . . 738
9.486 metadata_entry Struct Reference . . . 739
9.487 MetaPayload Class Reference . . . 740
9.488 Metrology Class Reference . . . 741
9.489 MetrologyHandler Class Reference . . . 742
9.490 MetrologyLoop Class Reference . . . 743
9.491 MetrologyPIDLoop< T > Class Template Reference . . . 744
9.492 MidRangeProbes Class Reference . . . 745

9.492.1 Detailed Description . . . 745
9.493 MirrorDistance Struct Reference . . . 746
9.493.1 Detailed Description . . . 746
9.494 AcqMiscStatus::misc_status Struct Reference . . . 747
9.495 MiscInfo Class Reference . . . 748
9.496 ModalFitData Class Reference . . . 749
9.497 ModalMatrix Class Reference . . . 750
9.498 modbus_mapping_t Struct Reference . . . 751
9.499 MODBUSComm Class Reference . . . 752
9.499.1 Detailed Description . . . 752
9.499.2 Member Function Documentation . . . 752
9.499.2.1 Connect . . . 752
9.499.2.2 getValues . . . 753
9.499.2.3 setValues . . . 753
9.500 MODBUSTime Struct Reference . . . 754
9.501 modified_julian_day Class Reference . . . 755
9.502 MonitorConfig Class Reference . . . 756
9.503 MonitorDriver Class Reference . . . 757
9.503.1 Member Function Documentation . . . 757
9.503.1.1 process . . . 757
9.504 liborchestrate::MonitorSubsystem Class Reference . . . 758
9.505 MonPrintRegister Class Reference . . . 759
9.506 MonWriteRegister Class Reference . . . 760
9.507 MountModel Class Reference . . . 761
9.507.1 Member Function Documentation . . . 761
9.507.1.1 focus_model_low . . . 761
9.507.1.2 itf_to_rtf . . . 761
9.507.1.3 pointing_mount_model_offset . . . 761
9.507.1.4 rotateXY . . . 761
9.507.1.5 rtf_to_itf . . . 762
9.508 mountModel Class Reference . . . 763
9.508.1 Detailed Description . . . 763
9.509 MountModelException Class Reference . . . 764
9.510 MoveProbeParms Class Reference . . . 765
9.511 MoveProbes Class Reference . . . 766
9.511.1 Detailed Description . . . 766

9.512MoveProbes_Enc Class Reference . . . 767
9.512.1 Detailed Description . . . 767
9.513MoveProbes_RaDec Class Reference . . . 768
9.513.1 Detailed Description . . . 768
9.514moveRTF Class Reference . . . 769
9.514.1 Detailed Description . . . 769
9.514.2 Member Data Documentation . . . 769
9.514.2.1 description . . . 769
9.514.2.2 h . . . 770
9.514.2.3 pos_dict . . . 770
9.515tracker_handlers::moveRTF Class Reference . . . 771
9.515.1 Member Function Documentation . . . 771
9.515.1.1 process . . . 771
9.516MoveTo Class Reference . . . 772
9.516.1 Detailed Description . . . 772
9.517MoveToStructAz Class Reference . . . 773
9.517.1 Detailed Description . . . 773
9.518moveTSF Class Reference . . . 774
9.518.1 Detailed Description . . . 774
9.518.2 Member Data Documentation . . . 774
9.518.2.1 description . . . 774
9.518.2.2 h . . . 775
9.518.2.3 pos_dict . . . 775
9.519tracker_handlers::moveTSF Class Reference . . . 776
9.519.1 Member Function Documentation . . . 776
9.519.1.1 process . . . 776
9.520MsgAttribute Class Reference . . . 777
9.521tthread::mutex Class Reference . . . 778
9.521.1 Detailed Description . . . 778
9.521.2 Member Function Documentation . . . 778
9.521.2.1 lock . . . 778
9.521.2.2 try_lock . . . 778
9.521.2.3 unlock . . . 779
9.522Mux Class Reference . . . 780
9.523liborchestrate::Named Class Reference . . . 781
9.524namedClient Class Reference . . . 782

9.525namedConfig Class Reference . . . 783
9.525.1 Member Function Documentation . . . 783
9.525.1.1 print_summary . . . 783
9.525.1.2 set_options . . . 783
9.526NamedException Class Reference . . . 784
9.527namedHandler Class Reference . . . 785
9.527.1 Member Function Documentation . . . 785
9.527.1.1 process . . . 785
9.528util::NamedReader Class Reference . . . 786
9.529namedServer Class Reference . . . 787
9.530native_wrapper< Type > Class Template Reference . . . 788
9.531NoiseAreaData Class Reference . . . 789
9.532noncopyable Class Reference . . . 790
9.532.1 Detailed Description . . . 790
9.533NOOPHandler Class Reference . . . 791
9.533.1 Member Function Documentation . . . 791
9.533.1.1 process . . . 791
9.534tcs_photometry::Object Struct Reference . . . 792
9.534.1 Detailed Description . . . 792
9.535wfs_photometry::WFSCentroid::object_t Struct Reference . . . 794
9.536ObscGeom Class Reference . . . 795
9.537OCDStatus Class Reference . . . 796
9.538offset_fiducial Class Reference . . . 797
9.538.1 Detailed Description . . . 797
9.538.2 Member Function Documentation . . . 797
9.538.2.1 process . . . 797
9.539offset_focus Class Reference . . . 798
9.539.1 Detailed Description . . . 798
9.539.2 Member Function Documentation . . . 798
9.539.2.1 process . . . 798
9.540offset_itf Class Reference . . . 799
9.540.1 Detailed Description . . . 799
9.540.2 Member Function Documentation . . . 799
9.540.2.1 process . . . 799
9.541offset_probe Class Reference . . . 800
9.541.1 Detailed Description . . . 800

9.541.2 Member Function Documentation . . . 800
9.541.2.1 process . . . 800
9.542 offset_sky Class Reference . . . 801
9.542.1 Detailed Description . . . 801
9.542.2 Member Function Documentation . . . 801
9.542.2.1 process . . . 801
9.543 offset_tiptilt Class Reference . . . 802
9.543.1 Detailed Description . . . 802
9.543.2 Member Function Documentation . . . 802
9.543.2.1 process . . . 802
9.544 offset_trajectory Class Reference . . . 803
9.544.1 Detailed Description . . . 803
9.544.2 Member Function Documentation . . . 803
9.544.2.1 process . . . 803
9.545 OModalMatrix Class Reference . . . 804
9.546 OnLine Class Reference . . . 805
9.546.1 Detailed Description . . . 805
9.547 Open Class Reference . . . 806
9.547.1 Detailed Description . . . 806
9.548 camra_handlers::open_shutter Class Reference . . . 807
9.548.1 Member Function Documentation . . . 807
9.548.1.1 process . . . 807
9.549 open_shutter Class Reference . . . 808
9.549.1 Detailed Description . . . 808
9.550 OpenShutter Class Reference . . . 809
9.550.1 Detailed Description . . . 809
9.551 camra_handlers::override_ifu_associations Class Reference . . . 810
9.551.1 Member Function Documentation . . . 810
9.551.1.1 process . . . 810
9.552 override_ifu_associations Class Reference . . . 811
9.552.1 Detailed Description . . . 811
9.553 P Class Reference . . . 812
9.553.1 Member Function Documentation . . . 812
9.553.1.1 process . . . 812
9.554 ParmsIterator Class Reference . . . 813
9.554.1 Detailed Description . . . 813

9.555ParmsObject Class Reference . . . 814
9.555.1 Detailed Description . . . 814
9.556PASCameraClient Class Reference . . . 815
9.556.1 Detailed Description . . . 815
9.557pasConfig Class Reference . . . 816
9.557.1 Member Function Documentation . . . 816
9.557.1.1 print_summary . . . 816
9.557.1.2 set_options . . . 816
9.558pasDevicePipelineSubSystem Class Reference . . . 817
9.559PASException Class Reference . . . 819
9.560pasImageCleanerStandard Class Reference . . . 820
9.561pasImagePipelineSubSystem Class Reference . . . 821
9.561.1 Member Function Documentation . . . 825
9.561.1.1 addTCSDataToFITSFile . . . 825
9.562pasImageProcessorPhot Class Reference . . . 826
9.563pasImageProcessorWFS Class Reference . . . 827
9.564pasImageStorage Class Reference . . . 828
9.565pasImageStorageFITS Class Reference . . . 829
9.566pasPipelineSubSystem Class Reference . . . 830
9.567pasServer Class Reference . . . 832
9.568liborchestrate::PasSubsystem Class Reference . . . 833
9.569pasTTLaserLightSource Class Reference . . . 834
9.570Payload Class Reference . . . 835
9.570.1 Detailed Description . . . 835
9.570.2 Member Function Documentation . . . 836
9.570.2.1 data . . . 836
9.570.2.2 operator<< . . . 836
9.570.2.3 set_data . . . 837
9.570.2.4 set_parm . . . 837
9.570.2.5 size . . . 837
9.571pduConfig Class Reference . . . 838
9.571.1 Member Function Documentation . . . 838
9.571.1.1 set_options . . . 838
9.572PduOnLine Class Reference . . . 839
9.572.1 Detailed Description . . . 839
9.573pfip_clear_positions Class Reference . . . 840

9.573.1 Detailed Description . . . 840
9.573.2 Member Function Documentation . . . 840
9.573.2.1 process . . . 840
9.574 pfip_close_shutter Class Reference . . . 841
9.574.1 Detailed Description . . . 841
9.574.2 Member Function Documentation . . . 841
9.574.2.1 process . . . 841
9.575 pfip_coord Class Reference . . . 842
9.575.1 Detailed Description . . . 842
9.575.2 Constructor & Destructor Documentation . . . 842
9.575.2.1 pfip_coord . . . 842
9.575.3 Member Function Documentation . . . 842
9.575.3.1 telecen2probe . . . 842
9.576 PFIP_EPICSVariable Class Reference . . . 844
9.577 pfip_insert_acquisition_pickoff Class Reference . . . 845
9.577.1 Detailed Description . . . 845
9.577.2 Member Function Documentation . . . 845
9.577.2.1 process . . . 845
9.578 PFIP_Monitor Class Reference . . . 846
9.578.1 Detailed Description . . . 846
9.578.2 Constructor & Destructor Documentation . . . 846
9.578.2.1 PFIP_Monitor . . . 846
9.579 pfip_move_probes Class Reference . . . 847
9.579.1 Detailed Description . . . 847
9.579.2 Member Function Documentation . . . 847
9.579.2.1 process . . . 847
9.580 pfip_open_shutter Class Reference . . . 848
9.580.1 Detailed Description . . . 848
9.580.2 Member Function Documentation . . . 848
9.580.2.1 process . . . 848
9.581 pfip_withdraw_acquisition_pickoff Class Reference . . . 849
9.581.1 Detailed Description . . . 849
9.581.2 Member Function Documentation . . . 849
9.581.2.1 process . . . 849
9.582 PFIPClient Class Reference . . . 850
9.583 PFIPClientInstrumentHandler Class Reference . . . 851

9.584 PFIPConfig Class Reference . . . 852
9.584.1 Detailed Description . . . 852
9.584.2 Member Function Documentation . . . 854
9.584.2.1 print_summary . . . 854
9.584.2.2 set_options . . . 854
9.585 Ui::pfipControlTab Class Reference . . . 855
9.586 pfipControlTabUi Class Reference . . . 856
9.587 pasImagePipelineSubSystem::PFIPData Struct Reference . . . 857
9.588 Ui::pfipEnvironmentTab Class Reference . . . 858
9.589 pfipEnvironmentTabUi Class Reference . . . 859
9.590 PFIPException Class Reference . . . 860
9.591 PFIPHandler Class Reference . . . 861
9.591.1 Constructor & Destructor Documentation . . . 861
9.591.1.1 PFIPHandler . . . 861
9.592 PFIPInterrupt Class Reference . . . 862
9.592.1 Detailed Description . . . 862
9.593 PFIPLampsDB Class Reference . . . 863
9.594 PFIPModbusException Class Reference . . . 864
9.595 PFIPMonitorStatus Class Reference . . . 865
9.596 PFIPProbeInfo Class Reference . . . 866
9.596.1 Member Function Documentation . . . 867
9.596.1.1 IsProbeEnabled . . . 867
9.596.1.2 LogErrors . . . 867
9.596.1.3 WaitForProbeEnabled . . . 867
9.597 pfipServer Class Reference . . . 868
9.597.1 Detailed Description . . . 868
9.597.2 Constructor & Destructor Documentation . . . 868
9.597.2.1 pfipServer . . . 868
9.598 PFIPStatusActions Class Reference . . . 869
9.598.1 Constructor & Destructor Documentation . . . 869
9.598.1.1 PFIPStatusActions . . . 869
9.599 PFIPStatusBase Class Reference . . . 870
9.600 liborchestrate::PfipSubsystem Class Reference . . . 871
9.601 tcs_photometry::PhotData Struct Reference . . . 872
9.601.1 Detailed Description . . . 872
9.602 tcs_photometry::Photometry Class Reference . . . 873

9.602.1 Detailed Description . . . 873
9.602.2 Member Function Documentation . . . 875
9.602.2.1 aperture_phot . . . 875
9.602.2.2 assemble . . . 875
9.602.2.3 check_variance . . . 875
9.602.2.4 copy_square . . . 875
9.602.2.5 elliptical_annulus_mask . . . 875
9.602.2.6 elliptical_mask . . . 875
9.602.2.7 fast_aperture_phot . . . 876
9.602.2.8 fast_fixed_aperture_phot . . . 876
9.602.2.9 fast_kron_aperture_phot . . . 876
9.602.2.10 find_stars . . . 876
9.602.2.11 fit_bivariate_gaussian . . . 876
9.602.2.12 fit_bivariate_moffat . . . 876
9.602.2.13 fixed_aperture_phot . . . 876
9.602.2.14 Guider_photometry . . . 876
9.602.2.15 integrate_soft_e_mask_fixed . . . 877
9.602.2.16 kron_aperture_phot . . . 877
9.602.2.17 mark_pixels . . . 877
9.602.2.18 resetGuideRect . . . 877
9.602.2.19 setGuideRect . . . 877
9.602.2.20 setPixscale . . . 877
9.602.2.21 square_annulus_mask . . . 877
9.602.2.22 subtract_background_fit . . . 877
9.602.2.23 subtract_background_fit_annulus . . . 878
9.602.2.24 subtract_background_fit_outer_edge . . . 878
9.602.2.25 TTCAM_photometry . . . 878
9.603 PhotometryData Class Reference . . . 879
9.604 Ui::photometryOptionsDialog Class Reference . . . 880
9.605 photometryOptionsDialog Class Reference . . . 881
9.606 PIDHandler< T > Class Template Reference . . . 882
9.607 PIDLoop< M, distance > Class Template Reference . . . 883
9.607.1 Detailed Description . . . 883
9.607.2 Member Function Documentation . . . 884
9.607.2.1 load_measurement . . . 884
9.607.2.2 set_proportional . . . 884

9.608pingHandler Class Reference . . . 885
9.609PipelineConfig Class Reference . . . 886
9.609.1 Member Function Documentation . . . 887
9.609.1.1 set_options . . . 887
9.610PipelineCyclePipelineHandlerClass Class Reference . . . 888
9.610.1 Member Function Documentation . . . 888
9.610.1.1 process . . . 888
9.611PipelineOnLine Class Reference . . . 889
9.611.1 Detailed Description . . . 889
9.612PipelineRelinquishControlHandlerClass Class Reference . . . 890
9.612.1 Member Function Documentation . . . 890
9.612.1.1 process . . . 890
9.613PipelineSetCycleTimeHandlerClass Class Reference . . . 891
9.613.1 Member Function Documentation . . . 891
9.613.1.1 process . . . 891
9.614PipelineSetIntervalSpeedHandlerClass Class Reference . . . 892
9.614.1 Member Function Documentation . . . 892
9.614.1.1 process . . . 892
9.615PipelineStartPipelineHandlerClass Class Reference . . . 893
9.615.1 Member Function Documentation . . . 893
9.615.1.1 process . . . 893
9.616PipelineStatus Class Reference . . . 894
9.617PipelineTakeControlHandlerClass Class Reference . . . 895
9.617.1 Member Function Documentation . . . 895
9.617.1.1 process . . . 895
9.618tcs_photometry::Pixel Struct Reference . . . 896
9.618.1 Detailed Description . . . 896
9.619Pixel Class Reference . . . 897
9.620Pixel_t Class Reference . . . 898
9.621ExposureInfo::PlateScale Struct Reference . . . 899
9.622plotEventsTabUi Class Reference . . . 900
9.623PlotterConfig Class Reference . . . 901
9.624PlotterDriver Class Reference . . . 902
9.624.1 Member Function Documentation . . . 902
9.624.1.1 process . . . 902
9.625liborchestrate::PlotterTcsSubsystem Class Reference . . . 903

9.626liborchestrate::PlotterTrackerSubsystem Class Reference . . . 904
9.627Point Class Reference . . . 905
9.627.1 Detailed Description . . . 905
9.627.2 Constructor & Destructor Documentation . . . 907
9.627.2.1 Point . . . 907
9.627.3 Member Function Documentation . . . 908
9.627.3.1 interp_ephem . . . 908
9.627.3.2 iterate . . . 908
9.627.3.3 optAz . . . 908
9.627.3.4 printEL . . . 908
9.627.3.5 printInversePointing . . . 908
9.627.4 Member Data Documentation . . . 909
9.627.4.1 tracks_ . . . 909
9.628point Struct Reference . . . 910
9.629Point_t Class Reference . . . 911
9.630pointing_coordinate Class Reference . . . 912
9.631pointing_ForceStatusEvents Class Reference . . . 913
9.632pointingConfig Class Reference . . . 914
9.632.1 Member Function Documentation . . . 914
9.632.1.1 print_summary . . . 914
9.632.1.2 set_options . . . 914
9.633pointingServer Class Reference . . . 915
9.634Pool< T > Class Template Reference . . . 916
9.634.1 Detailed Description . . . 916
9.634.2 Constructor & Destructor Documentation . . . 917
9.634.2.1 Pool . . . 917
9.635camra_handlers::populate_onewire_device_strings Class Reference . . . 918
9.635.1 Member Function Documentation . . . 918
9.635.1.1 process . . . 918
9.636populate_onewire_device_strings Class Reference . . . 919
9.636.1 Detailed Description . . . 919
9.637pos_fit Class Reference . . . 920
9.638fixed_point::fixed_internal::power10< N > Struct Template Reference . . . 921
9.639fixed_point::fixed_internal::power10< 0 > Struct Template Reference . . . 922
9.640fixed_point::fixed_internal::power10_aux< N, Even > Struct Template Reference . . . 923
9.641fixed_point::fixed_internal::power10_aux< N, true > Struct Template Reference . . . 924

9.642PowerACQCamera Class Reference . . . 925
9.642.1 Detailed Description . . . 925
9.643PowerBIBCamera Class Reference . . . 926
9.643.1 Detailed Description . . . 926
9.644PowerHPF_ACQCamera Class Reference . . . 927
9.644.1 Detailed Description . . . 927
9.645PowerHR_SASCamera Class Reference . . . 928
9.645.1 Detailed Description . . . 928
9.646PowerOffAll Class Reference . . . 929
9.646.1 Detailed Description . . . 929
9.647PowerOnAll Class Reference . . . 930
9.647.1 Detailed Description . . . 930
9.648PowerProbeCamera Class Reference . . . 931
9.648.1 Detailed Description . . . 931
9.649PowerPV_CWFSCamera Class Reference . . . 932
9.649.1 Detailed Description . . . 932
9.650PowerSBC1 Class Reference . . . 933
9.650.1 Detailed Description . . . 933
9.651PowerSBC2 Class Reference . . . 934
9.651.1 Detailed Description . . . 934
9.652PowerSwitch1 Class Reference . . . 935
9.652.1 Detailed Description . . . 935
9.653PowerSwitch2 Class Reference . . . 936
9.653.1 Detailed Description . . . 936
9.654PowerTTCamera Class Reference . . . 937
9.654.1 Detailed Description . . . 937
9.655Probe Class Reference . . . 938
9.655.1 Constructor & Destructor Documentation . . . 939
9.655.1.1 Probe . . . 939
9.655.2 Member Function Documentation . . . 940
9.655.2.1 CheckNewCurrentEncoderPositions . . . 940
9.655.2.2 ComputeMove . . . 940
9.655.3 Friends And Related Function Documentation . . . 940
9.655.3.1 ProbePosition::EncoderCountsToSphereRadians . . . 940
9.656ProbeAssembly Class Reference . . . 941
9.656.1 Member Function Documentation . . . 942

9.656.1.1 EnableDisableProbes . . . 942
9.656.1.2 GetAllProbeEncoderCounts . . . 942
9.656.1.3 GetProbeInfo . . . 942
9.656.1.4 Initialize . . . 942
9.656.1.5 ReadStatusValues . . . 942
9.656.1.6 UpdateProbePositions . . . 943
9.657 probeAssemblyScene Class Reference . . . 944
9.658 Ui::probeAssemblyTab Class Reference . . . 945
9.659 probeAssemblyTabUi Class Reference . . . 946
9.660 Ui::probeControlTab Class Reference . . . 947
9.661 probeControlTabUi Class Reference . . . 948
9.662 probeData Struct Reference . . . 949
9.663 Ui::probeGainControlTab Class Reference . . . 950
9.664 probeGainControlTabUi Class Reference . . . 951
9.665 ProbeInfo Class Reference . . . 952
9.666 ProbeItem Class Reference . . . 953
9.667 ProbePosition Class Reference . . . 954
9.667.1 Member Function Documentation . . . 954
9.667.1.1 GetEncoderCount . . . 954
9.667.1.2 SphereRadiansToEncoderCounts . . . 955
9.668 ProbeStatus Class Reference . . . 956
9.669 Ui::probeStatusTab Class Reference . . . 957
9.670 probeStatusTabUi Class Reference . . . 958
9.671 ProcessorConfig Class Reference . . . 959
9.671.1 Member Function Documentation . . . 960
9.671.1.1 set_options . . . 960
9.672 ProcessVariable< T > Class Template Reference . . . 961
9.673 Ui::productionTab Class Reference . . . 963
9.674 productionTabUi Class Reference . . . 964
9.675 proxy Class Reference . . . 966
9.675.1 Detailed Description . . . 966
9.676 Proxy Class Reference . . . 967
9.677 ProxyConfig Class Reference . . . 968
9.677.1 Member Function Documentation . . . 968
9.677.1.1 print_summary . . . 968
9.677.1.2 set_options . . . 968

9.678ProxyException Class Reference . . . 969
9.679ProxyHandler Class Reference . . . 970
9.679.1 Member Function Documentation . . . 970
9.679.1.1 process . . . 970
9.680PSFFitData Class Reference . . . 971
9.681PupilData Class Reference . . . 972
9.682QBlinkingLineEdit Class Reference . . . 973
9.683QBlinkingLineEditPlugin Class Reference . . . 974
9.684QGraphicsCircleItem Class Reference . . . 975
9.685QLedIndicator Class Reference . . . 976
9.686QLedIndicatorPlugin Class Reference . . . 978
9.687QSparkLine Class Reference . . . 979
9.688QSparkLinePlugin Class Reference . . . 980
9.689QTcsFrame Class Reference . . . 981
9.690QTcsFramePlugin Class Reference . . . 982
9.691QTcsProbeAssembly Class Reference . . . 983
9.692quadFit Class Reference . . . 985
9.693Quaternion< real > Class Template Reference . . . 986
9.694query_trajectory Class Reference . . . 987
9.694.1 Detailed Description . . . 987
9.694.2 Member Function Documentation . . . 987
9.694.2.1 process . . . 987
9.695tthread::ratio< N, D > Class Template Reference . . . 988
9.695.1 Detailed Description . . . 988
9.696camra_handlers::read_now Class Reference . . . 989
9.696.1 Member Function Documentation . . . 989
9.696.1.1 process . . . 989
9.697read_now Class Reference . . . 990
9.697.1 Detailed Description . . . 990
9.698ReadRegister Class Reference . . . 991
9.698.1 Detailed Description . . . 991
9.699Receiver Class Reference . . . 992
9.699.1 Detailed Description . . . 992
9.699.2 Constructor & Destructor Documentation . . . 992
9.699.2.1 Receiver . . . 992
9.699.3 Member Function Documentation . . . 992

9.699.3.1 registerHandler . . . 992
9.699.3.2 start . . . 993
9.699.3.3 stop . . . 993
9.700tthread::recursive_mutex Class Reference . . . 994
9.700.1 Detailed Description . . . 994
9.700.2 Member Function Documentation . . . 994
9.700.2.1 lock . . . 994
9.700.2.2 try_lock . . . 994
9.700.2.3 unlock . . . 995
9.701wfs_photometry::WFSCentroid::refList_t Struct Reference . . . 996
9.702RelinquishControl Class Reference . . . 997
9.702.1 Detailed Description . . . 997
9.703Reset Class Reference . . . 998
9.703.1 Detailed Description . . . 998
9.704camra_handlers::reset Class Reference . . . 999
9.704.1 Member Function Documentation . . . 999
9.704.1.1 process . . . 999
9.705reset Class Reference . . . 1000
9.705.1 Detailed Description . . . 1000
9.705.2 Member Function Documentation . . . 1000
9.705.2.1 process . . . 1000
9.706camra_handlers::reset_clock_voltages Class Reference . . . 1001
9.706.1 Member Function Documentation . . . 1001
9.706.1.1 process . . . 1001
9.707reset_clock_voltages Class Reference . . . 1002
9.707.1 Detailed Description . . . 1002
9.708reset_focus_offset Class Reference . . . 1003
9.708.1 Detailed Description . . . 1003
9.708.2 Member Function Documentation . . . 1003
9.708.2.1 process . . . 1003
9.709reset_itf_offset Class Reference . . . 1004
9.709.1 Detailed Description . . . 1004
9.709.2 Member Function Documentation . . . 1004
9.709.2.1 process . . . 1004
9.710camra_handlers::reset_mux Class Reference . . . 1005
9.710.1 Member Function Documentation . . . 1005

9.710.1.1 process . . . 1005
9.711 reset_mux Class Reference . . . 1006
9.711.1 Detailed Description . . . 1006
9.712 reset_radec_offset Class Reference . . . 1007
9.712.1 Detailed Description . . . 1007
9.712.2 Member Function Documentation . . . 1007
9.712.2.1 process . . . 1007
9.713 reset_sky_offset Class Reference . . . 1008
9.713.1 Detailed Description . . . 1008
9.713.2 Member Function Documentation . . . 1008
9.713.2.1 process . . . 1008
9.714 camra_handlers::reset_tcs Class Reference . . . 1009
9.714.1 Member Function Documentation . . . 1009
9.714.1.1 process . . . 1009
9.715 reset_tcs Class Reference . . . 1010
9.715.1 Detailed Description . . . 1010
9.716 reset_tiptilt_offset Class Reference . . . 1011
9.716.1 Detailed Description . . . 1011
9.716.2 Member Function Documentation . . . 1011
9.716.2.1 process . . . 1011
9.717 RetractACQCameraMirror Class Reference . . . 1012
9.717.1 Detailed Description . . . 1012
9.718 RetractCWFSCameraMirror Class Reference . . . 1013
9.718.1 Detailed Description . . . 1013
9.719 ExposureInfo::ROI Struct Reference . . . 1014
9.720 RowDispatcher Class Reference . . . 1015
9.720.1 Detailed Description . . . 1015
9.721 rtf_coordinate Class Reference . . . 1016
9.721.1 Detailed Description . . . 1016
9.722 sampleConfig Class Reference . . . 1017
9.722.1 Member Function Documentation . . . 1017
9.722.1.1 print_summary . . . 1017
9.722.1.2 set_options . . . 1017
9.723 SampleHandlerClass Class Reference . . . 1018
9.723.1 Member Function Documentation . . . 1018
9.723.1.1 process . . . 1018

9.724 sampleServer Class Reference . . . 1019
9.725 pasDevicePipelineSubSystem::SAMSData Struct Reference . . . 1020
9.726 SAMSData Class Reference . . . 1021
9.727 Sender Class Reference . . . 1022
9.727.1 Detailed Description . . . 1022
9.727.2 Constructor & Destructor Documentation . . . 1022
9.727.2.1 Sender . . . 1022
9.727.3 Member Function Documentation . . . 1022
9.727.3.1 send . . . 1022
9.727.3.2 send_async . . . 1023
9.727.3.3 wait_for . . . 1023
9.728 ExposureInfo::Sensor Struct Reference . . . 1024
9.729 ser7_table_entry Struct Reference . . . 1025
9.730 Server Class Reference . . . 1026
9.730.1 Detailed Description . . . 1026
9.731 ServerHandler Class Reference . . . 1027
9.731.1 Member Function Documentation . . . 1027
9.731.1.1 process . . . 1027
9.732 set Class Reference . . . 1028
9.732.1 Member Function Documentation . . . 1028
9.732.1.1 process . . . 1028
9.732.1.2 process . . . 1028
9.733 set_analysis_region Class Reference . . . 1029
9.733.1 Detailed Description . . . 1029
9.733.2 Member Function Documentation . . . 1029
9.733.2.1 process . . . 1029
9.734 camra_handlers::set_bias_offset Class Reference . . . 1030
9.734.1 Member Function Documentation . . . 1030
9.734.1.1 process . . . 1030
9.735 set_bias_offset Class Reference . . . 1031
9.735.1 Detailed Description . . . 1031
9.736 camra_handlers::set_ccd_temp Class Reference . . . 1032
9.736.1 Member Function Documentation . . . 1032
9.736.1.1 process . . . 1032
9.737 set_ccd_temp Class Reference . . . 1033
9.737.1 Detailed Description . . . 1033

9.738camra_handlers::set_ccd_temp_xfer Class Reference 1034
9.738.1 Member Function Documentation 1034
9.738.1.1 process 1034
9.739set_ccd_temp_xfer Class Reference 1035
9.739.1 Detailed Description 1035
9.740camra_handlers::set_clock_voltages Class Reference 1036
9.740.1 Member Function Documentation 1036
9.740.1.1 process 1036
9.741set_clock_voltages Class Reference 1037
9.741.1 Detailed Description 1037
9.742set_cryo_temp_xfer Class Reference 1038
9.742.1 Detailed Description 1038
9.743camra_handlers::set_cryo_temp_xfer Class Reference 1039
9.743.1 Member Function Documentation 1039
9.743.1.1 process 1039
9.744set_fiducial Class Reference 1040
9.744.1 Detailed Description 1040
9.744.2 Member Function Documentation 1040
9.744.2.1 process 1040
9.745set_filter Class Reference 1041
9.745.1 Detailed Description 1041
9.745.2 Member Function Documentation 1041
9.745.2.1 process 1041
9.746set_focus Class Reference 1042
9.746.1 Detailed Description 1042
9.746.2 Member Function Documentation 1042
9.746.2.1 process 1042
9.747set_focus_gain Class Reference 1043
9.747.1 Detailed Description 1043
9.747.2 Member Function Documentation 1043
9.747.2.1 process 1043
9.748set_focus_window_length Class Reference 1044
9.748.1 Detailed Description 1044
9.748.2 Member Function Documentation 1044
9.748.2.1 process 1044
9.749set_metadata Class Reference 1045

9.749.1 Detailed Description . . . 1045
9.749.2 Member Function Documentation . . . 1045
9.749.2.1 process . . . 1045
9.750 set_position Class Reference . . . 1046
9.750.1 Detailed Description . . . 1046
9.750.2 Member Function Documentation . . . 1046
9.750.2.1 process . . . 1046
9.751 set_tiptilt_gain Class Reference . . . 1047
9.751.1 Detailed Description . . . 1047
9.751.2 Member Function Documentation . . . 1047
9.751.2.1 process . . . 1047
9.752 set_tiptilt_window_length Class Reference . . . 1048
9.752.1 Detailed Description . . . 1048
9.752.2 Member Function Documentation . . . 1048
9.752.2.1 process . . . 1048
9.753 SetACQCameraFilter Class Reference . . . 1049
9.753.1 Detailed Description . . . 1049
9.754 SetACQCameraFocus Class Reference . . . 1050
9.754.1 Detailed Description . . . 1050
9.755 SetAnalysisRegion Class Reference . . . 1051
9.755.1 Detailed Description . . . 1051
9.756 SetBinning Class Reference . . . 1052
9.756.1 Detailed Description . . . 1052
9.757 setBufferSize Class Reference . . . 1053
9.757.1 Detailed Description . . . 1053
9.758 tracker_handlers::setBufferSize Class Reference . . . 1054
9.758.1 Member Function Documentation . . . 1054
9.758.1.1 process . . . 1054
9.759 SetCleanImages Class Reference . . . 1055
9.759.1 Detailed Description . . . 1055
9.760 SetCleaningBiasType Class Reference . . . 1056
9.760.1 Detailed Description . . . 1056
9.761 SetCycleTime Class Reference . . . 1057
9.761.1 Detailed Description . . . 1057
9.762 SetDirection Class Reference . . . 1058
9.762.1 Detailed Description . . . 1058

9.763 SetENWAFilter Class Reference . . . 1059
9.763.1 Detailed Description . . . 1059
9.764 SetExposuresPerCycle Class Reference . . . 1060
9.764.1 Detailed Description . . . 1060
9.765 SetExposureTime Class Reference . . . 1061
9.765.1 Detailed Description . . . 1061
9.766 SetFilterExposureTimeFactor Class Reference . . . 1062
9.766.1 Detailed Description . . . 1062
9.767 SetFilterSequence Class Reference . . . 1063
9.767.1 Detailed Description . . . 1063
9.768 SetFilterZeroPoint Class Reference . . . 1064
9.768.1 Detailed Description . . . 1064
9.769 SetGROC Class Reference . . . 1065
9.770 SetGuideCameraFilter Class Reference . . . 1066
9.770.1 Detailed Description . . . 1066
9.771 SetImageType Class Reference . . . 1067
9.771.1 Detailed Description . . . 1067
9.772 SetIntervalSpeed Class Reference . . . 1068
9.772.1 Detailed Description . . . 1068
9.773 SetLightSourcePower Class Reference . . . 1069
9.773.1 Detailed Description . . . 1069
9.774 SetLightSourcePowerHoldTime Class Reference . . . 1070
9.774.1 Detailed Description . . . 1070
9.775 SetLightSourcePowerLevel Class Reference . . . 1071
9.775.1 Detailed Description . . . 1071
9.776 SetNumFlushes Class Reference . . . 1072
9.776.1 Detailed Description . . . 1072
9.777 SetObjectAndMagnitudes Class Reference . . . 1073
9.777.1 Detailed Description . . . 1073
9.778 SetOverScan Class Reference . . . 1074
9.778.1 Detailed Description . . . 1074
9.779 SetPhotPhotometryOptions Class Reference . . . 1075
9.779.1 Detailed Description . . . 1075
9.780 SetPixelFormat Class Reference . . . 1076
9.780.1 Detailed Description . . . 1076
9.781 SetPower Class Reference . . . 1077

9.781.1 Detailed Description . . . 1077
9.782 SetProbeGains Class Reference . . . 1078
9.782.1 Detailed Description . . . 1078
9.783 SetProcessImages Class Reference . . . 1079
9.783.1 Detailed Description . . . 1079
9.784 SetProcessingType Class Reference . . . 1080
9.784.1 Detailed Description . . . 1080
9.785 SetReadoutSpeed Class Reference . . . 1081
9.785.1 Detailed Description . . . 1081
9.786 SetRegionOfInterest Class Reference . . . 1082
9.786.1 Detailed Description . . . 1082
9.787 SetStoreImages Class Reference . . . 1083
9.787.1 Detailed Description . . . 1083
9.788 SetTargetTemperature Class Reference . . . 1084
9.788.1 Detailed Description . . . 1084
9.789 setTraj Class Reference . . . 1085
9.789.1 Detailed Description . . . 1085
9.790 tracker_handlers::setTraj Class Reference . . . 1086
9.790.1 Member Function Documentation . . . 1086
9.790.1.1 process . . . 1086
9.791 SetVelocity Class Reference . . . 1087
9.791.1 Detailed Description . . . 1087
9.792 SetWFSPhotometryOptions Class Reference . . . 1088
9.792.1 Detailed Description . . . 1088
9.793 SHGrid Class Reference . . . 1089
9.794 ShotPoint Class Reference . . . 1090
9.795 SHTrackerData Class Reference . . . 1091
9.796 Shutter Class Reference . . . 1092
9.797 shutter_Abort Class Reference . . . 1093
9.797.1 Detailed Description . . . 1093
9.798 shutter_Close Class Reference . . . 1094
9.798.1 Detailed Description . . . 1094
9.799 shutter_IssuePMACCmd Class Reference . . . 1095
9.799.1 Detailed Description . . . 1095
9.800 shutter_OnLine Class Reference . . . 1096
9.800.1 Detailed Description . . . 1096

9.801 shutter_Open Class Reference . . . 1097
9.801.1 Detailed Description . . . 1097
9.802 shutter_Reset Class Reference . . . 1098
9.802.1 Detailed Description . . . 1098
9.803 shutter_SetPower Class Reference . . . 1099
9.803.1 Detailed Description . . . 1099
9.804 shutter_Status Class Reference . . . 1100
9.804.1 Detailed Description . . . 1100
9.805 AcqShutterStatus::shutter_status Struct Reference . . . 1101
9.806 shutter_Stop Class Reference . . . 1102
9.806.1 Detailed Description . . . 1102
9.807 ShutterConfig Class Reference . . . 1103
9.807.1 Member Function Documentation . . . 1103
9.807.1.1 set_options . . . 1103
9.808 ShutterInfo Class Reference . . . 1104
9.809 ShutterSetPowerHandlerClass Class Reference . . . 1105
9.809.1 Member Function Documentation . . . 1105
9.809.1.1 process . . . 1105
9.810 ShutterStatus Class Reference . . . 1106
9.811 ShutterStatusHandlerClass Class Reference . . . 1107
9.811.1 Member Function Documentation . . . 1107
9.811.1.1 process . . . 1107
9.812 Ui::shutterTab Class Reference . . . 1108
9.813 shutterTabUi Class Reference . . . 1109
9.814 SlopeMatrix Class Reference . . . 1110
9.815 sockcmd_reg_id_t Struct Reference . . . 1111
9.815.1 Detailed Description . . . 1111
9.815.2 Member Data Documentation . . . 1111
9.815.2.1 id . . . 1111
9.815.2.2 is_hld . . . 1111
9.816 sockcmd_t Struct Reference . . . 1112
9.816.1 Detailed Description . . . 1112
9.816.2 Member Data Documentation . . . 1112
9.816.2.1 cmd_code . . . 1112
9.816.2.2 reg_id . . . 1112
9.816.2.3 value . . . 1112

9.817 Socket Class Reference . . . 1113
9.817.1 Detailed Description . . . 1113
9.818 SocketException Class Reference . . . 1114
9.818.1 Detailed Description . . . 1114
9.819 tcsGui::soundObject Class Reference . . . 1115
9.820 tcsdb::source_descriptor Class Reference . . . 1116
9.821 pytcs::tcsdb::source_descriptor Class Reference . . . 1117
9.822 hetlib::spectrograph::Spectrograph Class Reference . . . 1118
9.822.1 Member Function Documentation . . . 1118
9.822.1.1 init_FmKeyStr . . . 1118
9.822.1.2 init_FmKeyStr . . . 1119
9.822.1.3 mkDisplay . . . 1119
9.822.1.4 mkDisplay . . . 1119
9.822.1.5 Pad . . . 1119
9.822.1.6 Pad . . . 1119
9.822.1.7 parse_KeyStr . . . 1119
9.822.1.8 parse_KeyStr . . . 1119
9.822.1.9 PowerStr . . . 1119
9.822.1.10 PowerStr . . . 1120
9.823 SpectrographConfig Class Reference . . . 1121
9.823.1 Detailed Description . . . 1121
9.823.2 Member Function Documentation . . . 1122
9.823.2.1 print_summary . . . 1122
9.823.2.2 set_options . . . 1122
9.823.3 Member Data Documentation . . . 1122
9.823.3.1 spectrograph_id . . . 1122
9.824 hetlib::spectrograph::SpectrographList Class Reference . . . 1123
9.824.1 Detailed Description . . . 1123
9.824.2 Member Function Documentation . . . 1123
9.824.2.1 find . . . 1123
9.824.2.2 find . . . 1123
9.824.2.3 sortSpect . . . 1123
9.824.2.4 sortSpect . . . 1123
9.825 HETillumination::Spidermotion Struct Reference . . . 1124
9.826 fixed_point::fixed_internal::square< N > Struct Template Reference . . . 1125
9.827 start Class Reference . . . 1126

9.827.1 Detailed Description . . . 1126
9.827.2 Member Function Documentation . . . 1126
9.827.2.1 process . . . 1126
9.828 StartPipeline Class Reference . . . 1127
9.828.1 Detailed Description . . . 1127
9.829 StartSendingImages Class Reference . . . 1128
9.829.1 Detailed Description . . . 1128
9.830 StaticConfig Struct Reference . . . 1129
9.831 legacyWeatherSubSystem::statistic Struct Reference . . . 1131
9.832 WeatherData::statistic Struct Reference . . . 1132
9.833 PipelineStatus::status Struct Reference . . . 1133
9.834 DomeStatus::status Struct Reference . . . 1134
9.835 Status Class Reference . . . 1135
9.835.1 Detailed Description . . . 1135
9.836 WeatherData::status Struct Reference . . . 1136
9.837 SAMSData::status Struct Reference . . . 1137
9.838 legacyWeatherSubSystem::status Struct Reference . . . 1138
9.839 PFIPStatusBase::status Struct Reference . . . 1139
9.840 APCData::status Struct Reference . . . 1140
9.841 TrackerStatus::status Struct Reference . . . 1141
9.842 OCDStatus::status Struct Reference . . . 1147
9.843 ThermoCubeStatus::status Struct Reference . . . 1148
9.844 ShutterStatus::status Struct Reference . . . 1150
9.845 TrackerPosition::status Struct Reference . . . 1151
9.846 legacyShutterSubSystem::status Struct Reference . . . 1152
9.847 VirusMonitorStatus::status Struct Reference . . . 1153
9.848 PFIPMonitorStatus::status Struct Reference . . . 1154
9.849 DMIData::status Struct Reference . . . 1155
9.850 StructureStatus::status Struct Reference . . . 1156
9.851 legacyDomeSubSystem::status Struct Reference . . . 1157
9.852 IglooData::status Struct Reference . . . 1158
9.853 legacyStructureSubSystem::status Struct Reference . . . 1159
9.854 StatusBlock< T > Class Template Reference . . . 1160
9.855 StatusData Class Reference . . . 1161
9.856 StatusHandler Class Reference . . . 1162
9.856.1 Detailed Description . . . 1162

9.856.2 Member Function Documentation . . . 1162
9.856.2.1 process . . . 1162
9.857 stencil< T > Class Template Reference . . . 1163
9.858 Stop Class Reference . . . 1164
9.858.1 Detailed Description . . . 1164
9.859 stop Class Reference . . . 1165
9.859.1 Detailed Description . . . 1165
9.859.2 Member Function Documentation . . . 1165
9.859.2.1 process . . . 1165
9.860 StopPipeline Class Reference . . . 1166
9.860.1 Detailed Description . . . 1166
9.861 StopSendingImages Class Reference . . . 1167
9.861.1 Detailed Description . . . 1167
9.862 Stow Class Reference . . . 1168
9.862.1 Detailed Description . . . 1168
9.863 structure_MoveToCCAS Class Reference . . . 1169
9.864 structure_MoveToStructAz Class Reference . . . 1170
9.864.1 Detailed Description . . . 1170
9.865 structure_MoveToTrueAz Class Reference . . . 1171
9.865.1 Detailed Description . . . 1171
9.866 structure_OnLine Class Reference . . . 1172
9.867 structure_Status Class Reference . . . 1173
9.867.1 Detailed Description . . . 1173
9.868 structure_Stop Class Reference . . . 1174
9.869 structure_Stow Class Reference . . . 1175
9.870 StructureClient Class Reference . . . 1176
9.871 StructureConfig Class Reference . . . 1177
9.871.1 Member Function Documentation . . . 1177
9.871.1.1 set_options . . . 1177
9.872 StructureModel Class Reference . . . 1178
9.873 StructureMoveToCCASHandlerClass Class Reference . . . 1179
9.873.1 Member Function Documentation . . . 1179
9.873.1.1 process . . . 1179
9.874 StructureMoveToStructAzHandlerClass Class Reference . . . 1180
9.874.1 Member Function Documentation . . . 1180
9.874.1.1 process . . . 1180

9.875StructureMoveToTrueAzHandlerClass Class Reference 1181
9.875.1 Member Function Documentation 1181
9.875.1.1 process 1181
9.876StructureOnLineHandlerClass Class Reference 1182
9.876.1 Member Function Documentation 1182
9.876.1.1 process 1182
9.877StructureStatus Class Reference 1183
9.878StructureStatusHandlerClass Class Reference 1184
9.878.1 Member Function Documentation 1184
9.878.1.1 process 1184
9.879StructureStowHandlerClass Class Reference 1185
9.879.1 Member Function Documentation 1185
9.879.1.1 process 1185
9.880Ui::structureTab Class Reference 1186
9.881structureTabUi Class Reference 1187
9.882SubSystem Class Reference 1188
9.883subSystem Class Reference 1190
9.884liborchestrate::Subsystem Class Reference 1191
9.885SubSystemConfig Class Reference 1192
9.885.1 Member Function Documentation 1192
9.885.1.1 print_summary 1192
9.885.1.2 set_options 1192
9.886SubSystemHandler Class Reference 1194
9.886.1 Member Function Documentation 1194
9.886.1.1 process 1194
9.887liborchestrate::SubsystemManager Class Reference 1195
9.888pytcs::tcsdb::system_descriptor Class Reference 1196
9.889tcsdb::system_descriptor Class Reference 1197
9.890apcTabUi::tabData Class Reference 1198
9.891TakeControl Class Reference 1199
9.891.1 Detailed Description 1199
9.892TakeExposure Class Reference 1200
9.892.1 Detailed Description 1200
9.893Tap Class Reference 1201
9.894TapHandler Class Reference 1202
9.894.1 Member Function Documentation 1202

9.894.1.1 process . . . 1202
9.895 TCSApp< Subsys, Config > Class Template Reference . . . 1203
9.896 TCSBaseClass Class Reference . . . 1204
9.896.1 Detailed Description . . . 1204
9.897 TCSBaseException Class Reference . . . 1205
9.898 TCSClient Class Reference . . . 1206
9.899 tcsCmd Class Reference . . . 1207
9.900 TCSConfig Class Reference . . . 1208
9.900.1 Detailed Description . . . 1208
9.900.2 Constructor & Destructor Documentation . . . 1210
9.900.2.1 TCSConfig . . . 1210
9.900.3 Member Function Documentation . . . 1210
9.900.3.1 configure . . . 1210
9.900.3.2 configure . . . 1211
9.900.3.3 print_summary . . . 1211
9.900.3.4 set_options . . . 1211
9.900.4 Member Data Documentation . . . 1211
9.900.4.1 config_root . . . 1211
9.900.4.2 debug . . . 1211
9.900.4.3 help . . . 1211
9.900.4.4 hostname . . . 1211
9.900.4.5 log_levels . . . 1212
9.900.4.6 name . . . 1212
9.900.4.7 program . . . 1212
9.900.4.8 version . . . 1212
9.901 TCSConfigException Class Reference . . . 1213
9.902 tcsConsole Class Reference . . . 1214
9.903 pasImagePipelineSubSystem::TCSData Struct Reference . . . 1215
9.904 tcsdb::tcsdb Class Reference . . . 1216
9.905 pytcs::tcsdb::tcsdb Class Reference . . . 1217
9.906 TCSDriver Class Reference . . . 1218
9.906.1 Detailed Description . . . 1218
9.906.2 Member Function Documentation . . . 1219
9.906.2.1 cancel_trajectory . . . 1219
9.906.2.2 clear_trajectories . . . 1219
9.906.2.3 get_radec_offset_register . . . 1219

9.906.2.4 go_next . . . 1219
9.906.2.5 reset_radec_offset_register . . . 1219
9.906.2.6 trajectory . . . 1219
9.906.2.7 trajectory_info . . . 1220
9.907TCSEvent::TCSEvent Class Reference . . . 1221
9.908tcs_event::TCSEvent Class Reference . . . 1222
9.908.1 Member Function Documentation . . . 1222
9.908.1.1 __init__ . . . 1222
9.908.1.2 connect . . . 1222
9.908.1.3 next . . . 1222
9.908.1.4 subscribe . . . 1222
9.909pytcs::tcs_event::TCSEvent Class Reference . . . 1223
9.909.1 Member Function Documentation . . . 1223
9.909.1.1 __init__ . . . 1223
9.909.1.2 connect . . . 1223
9.909.1.3 next . . . 1223
9.909.1.4 subscribe . . . 1223
9.910tcsEvents Class Reference . . . 1224
9.911TCSException Class Reference . . . 1225
9.912TcsException Class Reference . . . 1226
9.913tcsGui Class Reference . . . 1227
9.914tcsGuiThreadWorker Class Reference . . . 1233
9.915TCSHandler Class Reference . . . 1239
9.916TCSLog::TCSLog Class Reference . . . 1240
9.916.1 Member Function Documentation . . . 1240
9.916.1.1 __init__ . . . 1240
9.917pytcs::TCSLog::TCSLog Class Reference . . . 1241
9.917.1 Member Function Documentation . . . 1241
9.917.1.1 __init__ . . . 1241
9.918pytcs::TCSNamed::TCSNamed Class Reference . . . 1242
9.919TCSNamed::TCSNamed Class Reference . . . 1243
9.920TCSServerConfig Class Reference . . . 1244
9.920.1 Member Function Documentation . . . 1244
9.920.1.1 print_summary . . . 1244
9.920.1.2 set_options . . . 1244
9.921TCSSocket Class Reference . . . 1245

9.921.1 Member Function Documentation . . . 1246
9.921.1.1 connect . . . 1246
9.921.1.2 connected . . . 1246
9.921.1.3 disconnect . . . 1246
9.921.1.4 flush . . . 1246
9.921.1.5 receive . . . 1246
9.921.1.6 transmit . . . 1246
9.922pytcs::tcssubsystem::TCSSubSystem Class Reference . . . 1248
9.923TCSSubSystem Class Reference . . . 1249
9.923.1 Detailed Description . . . 1249
9.924tcssubsystem::TCSSubSystem Class Reference . . . 1250
9.925liborchestrate::TcsSubsystem Class Reference . . . 1251
9.926TCSTelnet Class Reference . . . 1252
9.927telescope Struct Reference . . . 1253
9.928TCSClient::TelescopeData Struct Reference . . . 1254
9.929Test Class Reference . . . 1255
9.929.1 Detailed Description . . . 1255
9.930test Class Reference . . . 1256
9.930.1 Member Function Documentation . . . 1256
9.930.1.1 process . . . 1256
9.931ThermoCubeStatus Class Reference . . . 1257
9.932tthread::thread Class Reference . . . 1258
9.932.1 Constructor & Destructor Documentation . . . 1259
9.932.1.1 thread . . . 1259
9.932.1.2 thread . . . 1259
9.932.1.3 thread . . . 1259
9.932.1.4 thread . . . 1260
9.932.1.5 thread . . . 1260
9.932.1.6 ∼thread . . . 1260
9.932.2 Member Function Documentation . . . 1260
9.932.2.1 detach . . . 1260
9.932.2.2 hardware_concurrency . . . 1261
9.932.2.3 join . . . 1261
9.932.2.4 joinable . . . 1261
9.932.2.5 native_handle . . . 1261
9.932.2.6 set_trampoline . . . 1261

9.932.2.7 thread_execute_once . . . 1261
9.933 tthread::ThreadPool Class Reference . . . 1262
9.933.1 Detailed Description . . . 1262
9.934 tthread::ThreadPoolException Class Reference . . . 1263
9.935 tthread::ThreadPoolWorker Struct Reference . . . 1264
9.936 threadsafe_deque< Entry > Class Template Reference . . . 1265
9.936.1 Detailed Description . . . 1265
9.937 threadsafe_list< Entry > Class Template Reference . . . 1266
9.937.1 Detailed Description . . . 1266
9.938 threadsafe_queue< Entry > Class Template Reference . . . 1267
9.938.1 Detailed Description . . . 1267
9.939 tthread::ThreadWorker Class Reference . . . 1268
9.939.1 Detailed Description . . . 1268
9.939.2 Constructor & Destructor Documentation . . . 1269
9.939.2.1 ThreadWorker . . . 1269
9.939.2.2 ThreadWorker . . . 1269
9.939.2.3 ThreadWorker . . . 1269
9.939.2.4 ThreadWorker . . . 1270
9.939.3 Member Function Documentation . . . 1270
9.939.3.1 is_running . . . 1270
9.939.3.2 run . . . 1270
9.939.3.3 set_period . . . 1270
9.939.3.4 set_trampoline . . . 1270
9.939.3.5 stop . . . 1270
9.939.3.6 thread_execute_periodic . . . 1270
9.940 TimeoutHandler Class Reference . . . 1272
9.940.1 Member Function Documentation . . . 1272
9.940.1.1 process . . . 1272
9.941 TimeoutNextMove Class Reference . . . 1273
9.941.1 Detailed Description . . . 1273
9.942 tthread::Timer Class Reference . . . 1274
9.942.1 Detailed Description . . . 1274
9.943 tracker_handlers::timeToIntercept Class Reference . . . 1275
9.943.1 Member Function Documentation . . . 1275
9.943.1.1 process . . . 1275
9.944 timeToIntercept Class Reference . . . 1276

9.944.1 Detailed Description . . . 1276
9.945tle_t Struct Reference . . . 1277
9.946TMCSState Class Reference . . . 1278
9.946.1 Detailed Description . . . 1278
9.947liborchestrate::TmcsSubsystem Class Reference . . . 1279
9.948toggle Class Reference . . . 1280
9.948.1 Member Function Documentation . . . 1280
9.948.1.1 process . . . 1280
9.949toggle_gui_token Class Reference . . . 1281
9.949.1 Detailed Description . . . 1281
9.949.2 Member Function Documentation . . . 1281
9.949.2.1 process . . . 1281
9.950TopConfig Class Reference . . . 1282
9.951TopDriver Class Reference . . . 1283
9.952tracCoord Struct Reference . . . 1284
9.953tracker_abort Class Reference . . . 1285
9.953.1 Detailed Description . . . 1285
9.953.2 Member Function Documentation . . . 1285
9.953.2.1 process . . . 1285
9.954tracker_coordinate Class Reference . . . 1286
9.954.1 Detailed Description . . . 1286
9.955tracker_init Class Reference . . . 1287
9.955.1 Detailed Description . . . 1287
9.955.2 Member Function Documentation . . . 1287
9.955.2.1 process . . . 1287
9.956tracker_move_angular Class Reference . . . 1288
9.956.1 Detailed Description . . . 1288
9.956.2 Member Function Documentation . . . 1288
9.956.2.1 process . . . 1288
9.957tracker_move_itf Class Reference . . . 1289
9.957.1 Detailed Description . . . 1289
9.957.2 Member Function Documentation . . . 1289
9.957.2.1 process . . . 1289
9.958tracker_move_tsf Class Reference . . . 1290
9.958.1 Detailed Description . . . 1290
9.958.2 Member Function Documentation . . . 1290

9.958.2.1 process . . . 1290
9.959 tracker_position Class Reference . . . 1291
9.959.1 Detailed Description . . . 1291
9.959.2 Member Function Documentation . . . 1291
9.959.2.1 process . . . 1291
9.960 GLTrackerDriver::tracker_position Struct Reference . . . 1292
9.961 TrackerClient Class Reference . . . 1293
9.962 TrackerConfig Class Reference . . . 1294
9.962.1 Member Function Documentation . . . 1294
9.962.1.1 print_summary . . . 1294
9.962.1.2 set_options . . . 1294
9.963 TrackerData Class Reference . . . 1295
9.964 TrackerDriver Class Reference . . . 1296
9.964.1 Detailed Description . . . 1296
9.964.2 Member Function Documentation . . . 1296
9.964.2.1 tracker_client . . . 1296
9.964.2.2 tracker_server . . . 1296
9.965 TrackerException Class Reference . . . 1297
9.966 TrackerHandler Class Reference . . . 1298
9.966.1 Detailed Description . . . 1298
9.967 HETillumination::Trackermotion Struct Reference . . . 1299
9.968 TrackerPosition Class Reference . . . 1300
9.969 TrackerStatus Class Reference . . . 1301
9.970 Ui::trackerStatusTab Class Reference . . . 1302
9.971 trackerStatusTabUi Class Reference . . . 1303
9.972 liborchestrate::TrackerSubsystem Class Reference . . . 1304
9.973 Ui::trackerTab Class Reference . . . 1305
9.974 trackerTabUi Class Reference . . . 1306
9.975 traj_class::Trajectory Class Reference . . . 1307
9.975.1 Detailed Description . . . 1307
9.975.2 Member Function Documentation . . . 1307
9.975.2.1 __init__ . . . 1307
9.976 Trajectory Class Reference . . . 1308
9.977 TrajectoryException Class Reference . . . 1309
9.978 traj_class::TrajectoryList Class Reference . . . 1310
9.978.1 Detailed Description . . . 1310

9.979 Ui::trajectoryTab Class Reference . . . 1311
9.980 trajectoryTabUi Class Reference . . . 1312
9.981 tsf_coordinate Class Reference . . . 1313
9.981.1 Detailed Description . . . 1313
9.982 tcs_photometry::TTCAMMetrologyData Struct Reference . . . 1314
9.983 tune< T > Class Template Reference . . . 1315
9.983.1 Detailed Description . . . 1315
9.983.2 Member Function Documentation . . . 1315
9.983.2.1 process . . . 1315
9.984 turn_off_leds Class Reference . . . 1316
9.984.1 Detailed Description . . . 1316
9.985 camra_handlers::turn_off_leds Class Reference . . . 1317
9.985.1 Member Function Documentation . . . 1317
9.985.1.1 process . . . 1317
9.986 Ui_aboutDialog Class Reference . . . 1318
9.987 Ui_acqCamStatusTab Class Reference . . . 1319
9.988 Ui_alertsDialog Class Reference . . . 1321
9.989 Ui_apcTab Class Reference . . . 1322
9.990 Ui_cameraControlTab Class Reference . . . 1323
9.991 Ui_cameraImageTab Class Reference . . . 1325
9.992 Ui_cameraMetrologyControlTab Class Reference . . . 1327
9.993 Ui_cameraMetrologyDataTab Class Reference . . . 1328
9.994 Ui_configureSystemsDialog Class Reference . . . 1329
9.995 Ui_configureTabsDialog Class Reference . . . 1330
9.996 Ui_confirmOffsetDialog Class Reference . . . 1331
9.997 Ui_dmiTab Class Reference . . . 1332
9.998 Ui_domeTab Class Reference . . . 1334
9.999 Ui_editPreferencesDialog Class Reference . . . 1336
9.1000 Ui_eventMonitorTab Class Reference . . . 1338
9.1001 Ui_fcuControlTab Class Reference . . . 1339
9.1002 Ui_fcuStatusTab Class Reference . . . 1341
9.1003 Ui_fiducialDialog Class Reference . . . 1343
9.1004 Ui_focalPlaneShutterTab Class Reference . . . 1344
9.1005 Ui_guideBoxDialog Class Reference . . . 1345
9.1006 Ui_iglooTab Class Reference . . . 1346
9.1007 Ui_legacyTab Class Reference . . . 1347

9.1008 Ui_loadMarkersDialog Class Reference 1351
9.1009 Ui_mainWindow Class Reference 1352
9.1010 Ui_pfipControlTab Class Reference 1354
9.1011 Ui_pfipEnvironmentTab Class Reference 1356
9.1012 Ui_photometryOptionsDialog Class Reference 1358
9.1013 Ui_probeAssemblyTab Class Reference 1359
9.1014 Ui_probeControlTab Class Reference 1360
9.1015 Ui_probeGainControlTab Class Reference 1364
9.1016 Ui_probeStatusTab Class Reference 1365
9.1017 Ui_productionTab Class Reference 1369
9.1018 Ui_shutterTab Class Reference 1377
9.1019 Ui_structureTab Class Reference 1378
9.1020 Ui_trackerStatusTab Class Reference 1379
9.1021 Ui_trackerTab Class Reference 1387
9.1022 Ui_trajectoryTab Class Reference 1394
9.1023 Ui_videoSwitchTab Class Reference 1398
9.1024 Ui_virusEnvironmentTab Class Reference 1399
9.1025 Ui_weatherDataTab Class Reference 1405
9.1026 Ui_webBrowserTab Class Reference 1408
9.1027 UnhandledHandler Class Reference 1409
9.1027.1 Member Function Documentation 1409
9.1027.1.1 lprocess 1409
9.1028 Unimplemented Class Reference 1410
9.1029 UnknownException Class Reference 1411
9.1030 Unset_position Class Reference 1412
9.1030.1 Detailed Description 1412
9.1030.2 Member Function Documentation 1412
9.1030.2.1 lprocess 1412
9.1031 Update_setup Class Reference 1413
9.1031.1 Detailed Description 1413
9.1031.2 Member Function Documentation 1413
9.1031.2.1 lprocess 1413
9.1032 UptimeHandler Class Reference 1414
9.1032.1 Member Function Documentation 1414
9.1032.1.1 lprocess 1414
9.1033 Utc Class Reference 1415

9.1033.1 Detailed Description . . . 1415
9.1034 utc_iso8601 Class Reference . . . 1416
9.1035 UtilityGui Class Reference . . . 1417
9.1036 UtilityGuiThreadWorker Class Reference . . . 1419
9.1037 ValidPixelFormat Struct Reference . . . 1420
9.1038 wfs_photometry::WFSCentroid::vector_t Struct Reference . . . 1421
9.1039 Venc1PrintRegister Class Reference . . . 1422
9.1039.1 Detailed Description . . . 1422
9.1040 Venc1WriteRegister Class Reference . . . 1423
9.1040.1 Detailed Description . . . 1423
9.1041 Venc2PrintRegister Class Reference . . . 1424
9.1041.1 Detailed Description . . . 1424
9.1042 Venc2WriteRegister Class Reference . . . 1425
9.1042.1 Detailed Description . . . 1425
9.1043 VENCHandler Class Reference . . . 1426
9.1044 Ui::videoSwitchTab Class Reference . . . 1427
9.1045 VideoSwitchTabUi Class Reference . . . 1428
9.1046 VideoSwitchTabUiThreadWorker Class Reference . . . 1430
9.1047 VIRUS_Monitor Class Reference . . . 1431
9.1048 Ui::virusEnvironmentTab Class Reference . . . 1432
9.1049 VirusEnvironmentTabUi Class Reference . . . 1433
9.1050 VirusMonitorStatus Class Reference . . . 1434
9.1051 VirusEnvironmentTabUi::virusTempData Struct Reference . . . 1435
9.1052 camra_handlers::wait_for_readout Class Reference . . . 1436
9.1052.1 Member Function Documentation . . . 1436
9.1052.1.1 process . . . 1436
9.1053 wait_for_readout Class Reference . . . 1437
9.1053.1 Detailed Description . . . 1437
9.1054 wait_for_setup Class Reference . . . 1438
9.1054.1 Detailed Description . . . 1438
9.1054.2 Member Function Documentation . . . 1438
9.1054.2.1 process . . . 1438
9.1055 camra_handlers::wait_for_shutter_close Class Reference . . . 1439
9.1055.1 Member Function Documentation . . . 1439
9.1055.1.1 process . . . 1439
9.1056 wait_for_shutter_close Class Reference . . . 1440

9.1056.1 Detailed Description . . . 1440
9.1057 camra_handlers::wait_for_write Class Reference . . . 1441
9.1057.1 Member Function Documentation . . . 1441
9.1057.1.1 process . . . 1441
9.1058 wait_for_write Class Reference . . . 1442
9.1058.1 Detailed Description . . . 1442
9.1059 weather Class Reference . . . 1443
9.1060 weather_GetData Class Reference . . . 1444
9.1060.1 Detailed Description . . . 1444
9.1061 weather_OnLine Class Reference . . . 1445
9.1062 WeatherClient Class Reference . . . 1446
9.1063 WeatherConfig Class Reference . . . 1447
9.1063.1 Member Function Documentation . . . 1447
9.1063.1.1 set_options . . . 1447
9.1064 WeatherData Class Reference . . . 1448
9.1065 Ui::weatherDataTab Class Reference . . . 1449
9.1066 weatherDataTabUi Class Reference . . . 1450
9.1067 WeatherGetDataHandlerClass Class Reference . . . 1451
9.1067.1 Member Function Documentation . . . 1451
9.1067.1.1 process . . . 1451
9.1068 WeatherOnLine Class Reference . . . 1452
9.1068.1 Detailed Description . . . 1452
9.1069 WeatherOnLineHandlerClass Class Reference . . . 1453
9.1069.1 Member Function Documentation . . . 1453
9.1069.1.1 process . . . 1453
9.1070 Ui::webBrowserTab Class Reference . . . 1454
9.1071 webBrowserTabUi Class Reference . . . 1455
9.1072 webBrowserTabUiThreadWorker Class Reference . . . 1456
9.1073 wfs_photometry::WFSCentroid Class Reference . . . 1457
9.1074 wfsData Class Reference . . . 1458
9.1075 WFSException Class Reference . . . 1459
9.1076 wfs_photometry::WFSFit Class Reference . . . 1460
9.1077 wfs_photometry::WFSFitException Class Reference . . . 1461
9.1078 WFSHandler Class Reference . . . 1462
9.1079 wfslib Class Reference . . . 1463
9.1080 wfslib_het Class Reference . . . 1465

9.1081wfsLogger Class Reference 1468
9.1082WFSLoop Class Reference 1469
9.1083WFSToTCSLogLevelMap_t Struct Reference 1470
9.1084WriteOneWire Class Reference 1471
9.1085WriteOneWireConfig Class Reference 1472
9.1085.1Member Function Documentation 1472
9.1085.1.1print_summary 1472
9.1085.1.2set_options 1472
9.1086WriteRegister Class Reference 1473
9.1086.1Detailed Description 1473
9.1087xsky_find Class Reference 1474
9.1087.1Detailed Description 1474

10 File Documentation **1475**

10.1 het_lib/tthread/include/fast_mutex.h File Reference 1475
10.1.1 Detailed Description 1475
10.2 het_lib/tthread/include/tinythread.h File Reference 1476
10.2.1 Detailed Description 1476
10.2.2 Define Documentation 1477
10.2.2.1 _TTHREAD_DISABLE_ASSIGNMENT 1477
10.2.2.2 thread_local 1478

Chapter 1

Main Page

1.1 HET Processes

The Hobby-Eberly Telescope Control System consists of a set of distributed processes that control the operations of the Hobby-Eberly Telescope Wide Field Upgrade. The top level system processes are the TCS GUI, which provides a human interface to the telescope control system and the TCS Server, which coordinates the operations necessary to perform astronomical activities and monitoring.

- Telescope Control System (TCS)

The PFIP server provides a command api to control several different hardware interfaces -- the acquisition system camera and shutter, the guide probe assembly, the Facility Control Unit, and the monitoring system.

- Prime Focus Instrument Package Server (PFIPServer)

The PAS (Payload Alignment System) provides a control and image/data acquisition services layer for the HET cameras.

- Payload Alignment System (PAS)

The CCD Array Management and Recording Application (CAMRA) provides control and data acquisition services for the multiplexed ARC (Astronomical Research Camera, Inc.) CCD controller arrays.

- CCD Array Management and Recording Application (CAMRA)

The Tracker Server provides an interface between the HET subsystem communication protocol and the Tracker Control System protocol that controls the tracker. It also makes log entries into the HET Common Logger for the tracker.

- Tracker Server

The HET Log Server provides a central location where all system log messages can be stored.

- logsrvr

The legacy support subsystems, not included in the Wide Field Upgrade

- Legacy Servers Access System (Legacy)

1.2 HET Inter-Process Communication

The processes which makeup HET Telescope Control System communicate with each other using JSON-formatted strings. The contents of those strings are lists of parameter/value pairs. Aside from the legacy applications, the HET processes in the Wide Field Upgrade use message/reply and broadcast protocols. These are represented in the Message, Payload, EventSource, and Event classes. A process defines its set of available services using the MessageHandler class. Each type of operation supported has a handler. The message that goes to a handler comes in the form of the JSON-formatted string as either a Message or Payload object.

Each Message, Payload, or Event object has a base set of parameter/value pairs consisting of:

- the unix timestamp represented as a double
- the unix timestamp in string UTC form
- the process id that generates the object

When formatted as a string, the base message is something like the following:

```
{
        "pid":  7401,
        "unix_time":    1.4019955907941255695e+09,
        "unix_time_string":     "2014-06-05T19:13:10.794"
}
```

The remainder parameter/value pairs in a message are agreed-upon by both the receiver and sender. A parameter name is a string, which should be unique within the parameters of its message. A value can be one of several types:

- string
- double
- bool
- integer

1.3 HET Support Libraries

There are also support libraries for the basic infrastructure, such as

Coordinate transforms and mount modeling

- Coordinate Transforms and Structure Mount Model

Interprocess communication support

- Messaging Library

System-wide logging

- Common Client-Side Logging

Common HET application classes

- Common Library

Threading

- TinyThread++ API Reference

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

- Payload Alignment System (APC) . . . 70
  - APC Message Handlers . . . 71
    - Image Pdu . . . 69
  - APC Utility Classes . . . 72
    - Configuration . . . 65
      - Outlet . . . 66
      - Apc . . . 67
    - Image pdu Sub-System . . . 68
- CCD Array Management and Recording Application (CAMRA) . . . 79
  - CAMRA Handlers . . . 74
  - CAMRA Classes . . . 80
    - CAMRA Configuration Classes . . . 73
    - CAMRA Hardware Classes . . . 75
      - Array sizing constants . . . 76
      - Controller initialization constants . . . 77
      - Amplifier related constants . . . 78
- Common Library . . . 85
  - TCS Fixed precision time . . . 81
  - HET Earth orientation prediction . . . 82
  - TCS Application Class . . . 83
  - TCS Base Class . . . 84
  - TCS Base Configuration . . . 86
  - TCS SubSystem Class . . . 88
- Coordinate Systems and Transformations . . . 87
- Coordinate Transforms and Structure Mount Model . . . 89
  - Coordinate Transform Library . . . 90
  - Structure Mount Model . . . 91
- Common Client-Side Logging . . . 92
- Messaging Library . . . 94
  - Remote signalling . . . 93
- TinyThread++ API Reference . . . 95

Dome . . . 98
Dome . . . 102
Legacy Servers Access System (Legacy) . . . 103
  Class Documentation . . . 104
    Configuration . . . 96
      Dome . . . 97
      Structure . . . 99
      Weather . . . 100
    Dome . . . 101
    Shutter . . . 105
    Structure . . . 107
    Weather . . . 109
Shutter . . . 106
Structure . . . 108
Weather . . . 110
Payload Alignment System (PAS) . . . 128
  PAS Message Handlers . . . 129
    Image Pipeline . . . 124
  PAS Utility Classes . . . 130
    Camera . . . 111
      AVT Camera . . . 112
      FLI Camera . . . 113
      PointGrey Camera . . . 114
      Simulated Camera . . . 115
    Image Processor . . . 116
      Photometry Image Processor . . . 125
      Wave Front Sensor Image Processor . . . 126
    Light Source . . . 117
      Tip Tilt Light Source . . . 131
    Configuration . . . 118
      Light Source . . . 119
      Outlet . . . 66
      Image Processor . . . 120
      Pipe Line . . . 121
      Apc . . . 67
    Image Storage . . . 122
      FITS Image Storage . . . 127
    Image Pipeline Sub-System . . . 123
Prime Focus Instrument Package Server (PFIPServer) . . . 135
  PFIP Configuration Settings . . . 137
  PFIPServer Classes . . . 138
    Acquisition and Guide Probe Assembly . . . 132
    FCUClasses . . . 133
      MODBUS Utility Data Structures . . . 134
    Probe Assembly classes . . . 142
    EPICSVariables . . . 143
  PFIP Interface Operations and Events . . . 139
Acquisition and Probe Assembly . . . 140
Facility Calibration Unit . . . 141
Sample Servers Access System (Sample) . . . 145
  Message Handler Documentation . . . 146

Class Documentation 147
Configuration 144
Pointing Servers Access System (Pointing) 149
Class Documentation 150
Configuration 148
Telescope Control System (TCS) 151
Tracker Server 168
Tracker Motion Control System (TMCS) API 152

Chapter 3

Namespace Index

3.1 Namespace List

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

equ_hor . . . 169
hetlib::hetutils . . . 170
hetlib::Points . . . 173
hetlib::tcsconfig . . . 175
hetlib::tcsutils . . . 176
hetutils . . . 177
Points . . . 180
pt_itf . . . 182
st3 . . . 184
tcsconfig . . . 185
tcsutils . . . 186
tthread . . . 188
tthread::chrono . . . 190
tthread::this_thread . . . 191

Chapter 4

Class Index

4.1 Class Hierarchy

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

_modbus . . . 193
_modbus_backend . . . 194
_modbus_rtu . . . 195
_modbus_tcp . . . 196
_modbus_tcp_pi . . . 197
_sft . . . 198
tthread::_thread_start_info . . . 201
    tthread::_thread_start_functor< T > . . . 199
    tthread::_thread_start_functor_arg< F, A > . . . 200
    tthread::_thread_start_member< 14463331 > . . . 202
    tthread::_thread_start_member_arg< 14463331, A > . . . 203
Abort . . . 204
abort . . . 206
abortMove . . . 207
VirusMonitorStatus::alarms . . . 234
PFIPMonitorStatus::alarms . . . 235
altaz_coordinate . . . 238
Amoeba< Func > . . . 239
AnalysisRegion . . . 242
tcs_photometry::AperturePhotResults . . . 254
apply_offset . . . 255
tcsdb::attr_descriptor . . . 261
pytcs::tcsdb::attr_descriptor . . . 262
averager< Type > . . . 263
AVT . . . 264
bestDistance . . . 266
ExposureInfo::BINNING . . . 267
BuildPupilThreadArgs . . . 269
AcqCamStatus::camera_status . . . 273
CameraFLI::CameraData . . . 280
cameraImageMarker . . . 282
cameraImageScene . . . 283
CBUF< indexType, size, entryType > . . . 308

geHeaters::Ccd . . . 309
chunkHeader . . . 315
cJSON . . . 316
cJSON_Hooks . . . 317
clearError . . . 321
Close . . . 324
close_shutter . . . 326
coefs_t . . . 331
Command . . . 332
command_controller . . . 334
tthread::condition_variable . . . 335
tcs_photometry::ConfigureParams . . . 339
wfs_photometry::ConfigureParams . . . 340
cubFit . . . 352
AcqCWFSStatus::cwfs_status . . . 353
CyclePipeline . . . 355
mainWindow::DashBoardAlerts . . . 356
dataChunk . . . 358
deep_arg_t . . . 360
delta_time . . . 362
detachedTabWindow . . . 367
disable_pressure_gauges . . . 374
disable_shutter . . . 377
pasDevicePipelineSubSystem::DMIData . . . 380
Doh . . . 387
dome_Abort . . . 388
dome_GetPosition . . . 389
dome_Home . . . 390
dome_IssuePMACCmd . . . 391
dome_MoveTo . . . 392
dome_MoveToCCAS . . . 393
dome_OnLine . . . 394
dome_Reset . . . 395
dome_SetDirection . . . 396
dome_SetPower . . . 397
dome_SetVelocity . . . 398
dome_Status . . . 399
dome_Stop . . . 400
dome_Stow . . . 401
dome_Test . . . 402
droppablePlainTextEdit . . . 409
tthread::chrono::duration< _Rep, _Period > . . . 410
DynamicConfig . . . 411
enable_pressure_gauges . . . 414
enable_shutter . . . 418
EnableCCASDMI . . . 419
EnableSACDMI . . . 421
AcqENWAStatus::enwa_status . . . 424
Ephemeris . . . 426
EphList . . . 427
EphRec . . . 428
eq_coords . . . 436
replay::EventBuffer . . . 439
eventCmd . . . 440

EventHandler . . . 441
ICSConfig . . . 617
PFIPClient . . . 850
WeatherClient . . . 1446
EventList . . . 444
EventSink . . . 449
GLTrackerDriver . . . 583
MetrologyLoop . . . 743
DMILoop . . . 383
GuideLoop . . . 593
WFSLoop . . . 1469
MonitorDriver . . . 757
TCSDriver . . . 1218
TopDriver . . . 1283
EventSource . . . 451
ACQ_Camera . . . 211
ACQ_Shutter . . . 213
ACQ_SubSystem . . . 217
ACQCamInfo . . . 222
Alarm_Monitor . . . 233
PFIP_Monitor . . . 846
VIRUS_Monitor . . . 1431
Archon . . . 256
ArchonHandler . . . 257
test . . . 1256
ArrayType . . . 260
array_dbr_int_t . . . 258
array_dbr_long_t . . . 259
Bitfield . . . 268
Enum . . . 423
cameraImageTabUiThreadWorker . . . 289
CameraInfo . . . 290
camra_handlers::CAMRAHandler . . . 303
camra_handlers::abort . . . 205
camra_handlers::close_shutter . . . 325
camra_handlers::command_controller . . . 333
camra_handlers::disable_pressure_gauges . . . 373
camra_handlers::disable_shutter . . . 376
camra_handlers::enable_pressure_gauges . . . 415
camra_handlers::enable_shutter . . . 417
camra_handlers::expose . . . 455
camra_handlers::get_ccd_temp . . . 535
camra_handlers::get_clock_voltages . . . 537
camra_handlers::get_cryo_pressure . . . 540
camra_handlers::get_cryo_temp . . . 542
camra_handlers::get_exposure_info . . . 547
camra_handlers::get_hardware_status . . . 554
camra_handlers::get_heater_voltage . . . 556
camra_handlers::open_shutter . . . 807
camra_handlers::override_ifu_associations . . . 810
camra_handlers::populate_onewire_device_strings . . . 918
camra_handlers::read_now . . . 989

camra_handlers::reset . . . 999
camra_handlers::reset_clock_voltages . . . 1001
camra_handlers::reset_mux . . . 1005
camra_handlers::reset_tcs . . . 1009
camra_handlers::set_bias_offset . . . 1030
camra_handlers::set_ccd_temp . . . 1032
camra_handlers::set_ccd_temp_xfer . . . 1034
camra_handlers::set_clock_voltages . . . 1036
camra_handlers::set_cryo_temp_xfer . . . 1039
camra_handlers::turn_off_leds . . . 1317
camra_handlers::wait_for_readout . . . 1436
camra_handlers::wait_for_shutter_close . . . 1439
camra_handlers::wait_for_write . . . 1441
CAMRAConfig . . . 298
CAMRAHardware . . . 304
Config . . . 337
ConfigHandler . . . 338
get . . . 533
set . . . 1028
CWFSInfo . . . 354
DomeClient . . . 403
ENWAInfo . . . 425
EPICSComm . . . 429
EPICSHandle . . . 433
eventMonitorTabUiThreadWorker . . . 447
EventSourceRoot . . . 452
ICSDaemon< S > . . . 618
TCSSubSystem . . . 1249
apcCmd . . . 243
apcServer . . . 249
CAMRADriver . . . 300
EPICSRelayDriver . . . 435
GLTrackerDriver . . . 583
IglooSensors . . . 623
legacyServer . . . 681
LogRelayDriver . . . 709
MonitorDriver . . . 757
namedServer . . . 787
pasServer . . . 832
pfipServer . . . 868
PlotterDriver . . . 902
pointingServer . . . 915
Proxy . . . 967
sampleServer . . . 1019
SubSystem . . . 1188
apcPduSubSystem . . . 248
legacyDomeSubSystem . . . 677
legacyShutterSubSystem . . . 683
legacyStructureSubSystem . . . 686
legacyWeatherSubSystem . . . 692
pasPipelineSubSystem . . . 830
pasDevicePipelineSubSystem . . . 817
pasImagePipelineSubSystem . . . 821

tcsCmd . . . 1207
tcsConsole . . . 1214
TCSDriver . . . 1218
tcsGui . . . 1227
TopDriver . . . 1283
TrackerDriver . . . 1296
utilityGui . . . 1417
WriteOneWire . . . 1471
Exposure . . . 457
FCU_Shutter . . . 466
FCU_LDLS . . . 465
FCU_Slider . . . 467
FCU_State . . . 468
FCU_SubSystem . . . 472
FooBar . . . 524
HardwareElement . . . 603
Amp . . . 240
CCD . . . 310
Cntl . . . 329
Mux . . . 780
HETEarthOrientation . . . 608
ICSConfig . . . 617
IglooSensorsConnection . . . 625
IglooSensorsHandler . . . 627
check . . . 314
get . . . 533
set . . . 1028
toggle . . . 1280
Instrument . . . 665
loadedUi . . . 699
acqCamStatusTabUi . . . 225
apcTabUi . . . 252
cameraControlTabUi . . . 279
cameraImageTabUi . . . 287
cameraMetrologyControlTabUi . . . 292
cameraMetrologyDataTabUi . . . 294
dmiTabUi . . . 386
domeTabUi . . . 408
eventMonitorTabUi . . . 446
fcuControlTabUi . . . 479
fcuStatusTabUi . . . 504
focalPlaneShutterTabUi . . . 522
iglooTabUi . . . 632
legacyTabUi . . . 691
pfipControlTabUi . . . 856
pfipEnvironmentTabUi . . . 859
plotEventsTabUi . . . 900
probeAssemblyTabUi . . . 946
probeControlTabUi . . . 948
probeGainControlTabUi . . . 951
probeStatusTabUi . . . 958
productionTabUi . . . 964
shutterTabUi . . . 1109

structureTabUi . . . 1187
trackerStatusTabUi . . . 1303
trackerTabUi . . . 1306
trajectoryTabUi . . . 1312
videoSwitchTabUi . . . 1428
virusEnvironmentTabUi . . . 1433
weatherDataTabUi . . . 1450
webBrowserTabUi . . . 1455
loadedUi . . . 699
mainWindow . . . 714
mainWindow . . . 714
MAXONStatusInfo . . . 719
Metrology . . . 741
MetrologyHandler . . . 742
activate . . . 230
add_correction . . . 231
deactivate . . . 359
DMIHandler . . . 382
DMI_get_status . . . 379
force_state_event . . . 525
get_correction . . . 539
GuiderHandler . . . 595
clear_fiducial . . . 319
get_analysis_region . . . 534
get_fiducial . . . 549
offset_fiducial . . . 797
set_analysis_region . . . 1029
set_fiducial . . . 1040
inject_metrology . . . 664
PIDHandler< T > . . . 882
get_tuning< T > . . . 566
tune< T > . . . 1315
reset . . . 1000
start . . . 1126
stop . . . 1165
WFSHandler . . . 1462
center . . . 312
get_focus_gain . . . 550
get_focus_window_length . . . 552
get_tiptilt_gain . . . 563
get_tiptilt_window_length . . . 565
set_focus_gain . . . 1043
set_focus_window_length . . . 1044
set_tiptilt_gain . . . 1047
set_tiptilt_window_length . . . 1048
MetrologyLoop . . . 743
MiscInfo . . . 748
MODBUSComm . . . 752
MountModel . . . 761
namedClient . . . 782
namedHandler . . . 785
P . . . 812
PASCameraClient . . . 815

PFIPClient . . . 850
PFIPClient . . . 850
PFIPClientInstrumentHandler . . . 851
get_status . . . 562
offset_probe . . . 800
set_filter . . . 1041
set_position . . . 1046
unset_position . . . 1412
PFIPHandler . . . 861
AdjustDither . . . 232
BeginRangeProbes . . . 265
CloseShutter . . . 327
DeployACQCameraMirror . . . 363
DeployCWFSCamera . . . 364
DeployCWFSCameraMirror . . . 365
DeployPVCamera . . . 366
DisableProbes . . . 378
EnableProbes . . . 420
EndRangeProbes . . . 422
FCUHandler . . . 490
FCUCancelExposure . . . 475
FCUDeployHead . . . 480
FCUDisableAllLights . . . 481
FCUEnableLamps . . . 482
FCUEnableLightSources . . . 483
FCUExposeLDLS . . . 485
FCUExposeSliderA . . . 486
FCUExposeSliderB . . . 487
FCUGetCurrentExposureInfo . . . 488
FCUGetLastExposureInfo . . . 489
FCUInitLightSources . . . 492
FCUMoveLightBank . . . 493
FCUMoveLightGuide . . . 494
FCUMoveSlider . . . 495
FCUPrintRegister . . . 497
FCURetractHead . . . 498
FCUSetLampWarmUp . . . 499
FCUSetLDLSWarmUp . . . 500
FCUShutDown . . . 501
FCUWriteRegister . . . 505
GetACQSystemInfo . . . 567
GetProbeInfo . . . 575
GetProbePosition . . . 576
HomeProbes . . . 614
HomeShutter . . . 615
MidRangeProbes . . . 745
MoveProbes . . . 766
MoveProbes_Enc . . . 767
MoveProbes_RaDec . . . 768
OpenShutter . . . 809
PFIPInterrupt . . . 862
PowerACQCamera . . . 925
PowerBIBCamera . . . 926

PowerHPF_ACQCamera . . . 927
PowerHR_SASCamera . . . 928
PowerOffAll . . . 929
PowerOnAll . . . 930
PowerProbeCamera . . . 931
PowerPV_CWFSCamera . . . 932
PowerSBC1 . . . 933
PowerSBC2 . . . 934
PowerSwitch1 . . . 935
PowerSwitch2 . . . 936
PowerTTCamera . . . 937
ReadRegister . . . 991
RetractACQCameraMirror . . . 1012
RetractCWFSCameraMirror . . . 1013
SetACQCameraFilter . . . 1049
SetACQCameraFocus . . . 1050
SetENWAFilter . . . 1059
SetGuideCameraFilter . . . 1066
SetProbeGains . . . 1078
TakeExposure . . . 1200
TimeoutNextMove . . . 1273
VENCHandler . . . 1426
MonPrintRegister . . . 759
MonWriteRegister . . . 760
WriteRegister . . . 1473
PFIPLampsDB . . . 863
PFIPProbeInfo . . . 866
PFIPStatusActions . . . 869
Probe . . . 938
ProbeAssembly . . . 941
ProbeInfo . . . 952
ProbePosition . . . 954
ProcessVariable< T > . . . 961
ProxyHandler . . . 970
Shutter . . . 1092
ShutterInfo . . . 1104
StatusData . . . 1161
APCData . . . 246
DMIData . . . 381
DomeStatus . . . 405
FCUStatus . . . 502
IglooData . . . 621
OCDStatus . . . 796
PFIPMonitorStatus . . . 865
PFIPStatusBase . . . 870
AcqCamStatus . . . 223
AcqCWFSStatus . . . 226
AcqENWAStatus . . . 227
AcqMiscStatus . . . 228
AcqShutterStatus . . . 229
ProbeStatus . . . 956
PipelineStatus . . . 894
SAMSData . . . 1021

ShutterStatus . . . 1106
StructureStatus . . . 1183
ThermoCubeStatus . . . 1257
TrackerPosition . . . 1300
TrackerStatus . . . 1301
VirusMonitorStatus . . . 1434
WeatherData . . . 1448
StructureClient . . . 1176
Tap . . . 1201
TCSBaseClass . . . 1204
Camera . . . 271
CameraAVT . . . 274
CameraFLI . . . 281
CameraPGR . . . 295
CameraSim . . . 297
Filters . . . 511
ImageCleaner . . . 635
pasImageCleanerStandard . . . 820
ImageProcessor . . . 658
pasImageProcessorPhot . . . 826
pasImageProcessorWFS . . . 827
LightSource . . . 695
pasTTLaserLightSource . . . 834
pasImageStorage . . . 828
pasImageStorageFITS . . . 829
SubSystem . . . 1188
TCSClient . . . 1206
TCSClient . . . 1206
tcsGuiThreadWorker . . . 1233
TCSHandler . . . 1239
cancel_trajectory . . . 307
clear_trajectories . . . 320
correct_for_structure . . . 350
disable_rho . . . 375
enable_rho . . . 416
get_current_local_sidereal_time . . . 544
get_current_pointing . . . 545
get_current_trajectory_info . . . 546
get_focus_offset . . . 551
get_itf_offset . . . 557
get_metadata . . . 558
get_next_trajectory_info . . . 559
get_radec_offset . . . 560
get_sky_offset . . . 561
get_tiptilt_offset . . . 564
go_next . . . 585
load_trajectory . . . 698
offset_focus . . . 798
offset_itf . . . 799
offset_sky . . . 801
offset_tiptilt . . . 802
offset_trajectory . . . 803
pfip_clear_positions . . . 840

pfip_close_shutter . . . 841
pfip_insert_acquisition_pickoff . . . 845
pfip_move_probes . . . 847
pfip_open_shutter . . . 848
pfip_withdraw_acquisition_pickoff . . . 849
query_trajectory . . . 987
reset_focus_offset . . . 1003
reset_itf_offset . . . 1004
reset_radec_offset . . . 1007
reset_sky_offset . . . 1008
reset_tiptilt_offset . . . 1011
set_focus . . . 1042
set_metadata . . . 1045
toggle_gui_token . . . 1281
tracker_abort . . . 1285
tracker_init . . . 1287
tracker_move_angular . . . 1288
tracker_move_itf . . . 1289
tracker_move_tsf . . . 1290
tracker_position . . . 1291
update_setup . . . 1413
wait_for_setup . . . 1438
TMCSState . . . 1278
TrackerClient . . . 1293
TrackerHandler . . . 1298
tracker_handlers::abortMove . . . 208
tracker_handlers::clearError . . . 322
tracker_handlers::initTMCS . . . 663
tracker_handlers::moveRTF . . . 771
tracker_handlers::moveTSF . . . 776
tracker_handlers::setBufferSize . . . 1054
tracker_handlers::setTraj . . . 1086
tracker_handlers::timeToIntercept . . . 1275
Trajectory . . . 1308
Unimplemented . . . 1410
utilityGuiThreadWorker . . . 1419
videoSwitchTabUiThreadWorker . . . 1430
WeatherClient . . . 1446
WeatherClient . . . 1446
WeatherClient . . . 1446
wfslib_het . . . 1465
EventSourceSocket . . . 453
ExportData . . . 454
expose . . . 456
ExposureDescriptor . . . 458
ExposureInfo . . . 460
tthread::fast_mutex . . . 462
FCU_Lamp . . . 464
FCUStatus::fcu_status . . . 470
FCUCommunication . . . 477
FCUException . . . 484
FCUCommException . . . 476
FCUMutexException . . . 496

pasImagePipelineSubSystem::TCSData::fiducialObject . . . 508
ExposureInfo::FILTER_INFO . . . 509
AcqCamStatus::filter_status . . . 510
tcs_photometry::FitPars . . . 513
pasImagePipelineSubSystem::FITSMapEntry . . . 514
FitsOutWrapper . . . 515
fixed_point::FixedPoint< T, N > . . . 516
fixed_point::FixedPoint< int64_t, 9 > . . . 516
fptime . . . 529
PFIPStatusBase::flags . . . 518
flagValues . . . 519
focalPlaneShutterScene . . . 520
Foo . . . 523
ForceStatusEvent . . . 526
formatChunk . . . 528
fw102b::FW102B . . . 531
get_ccd_temp . . . 536
get_clock_voltages . . . 538
get_cryo_pressure . . . 541
get_cryo_temp . . . 543
get_exposure_info . . . 548
get_hardware_status . . . 553
get_heater_voltage . . . 555
GetAnalysisRegion . . . 568
GetExposureTime . . . 569
GetGain . . . 570
GetLightSourcePowerLevel . . . 571
GetOverScan . . . 572
GetPhotPhotometryOptions . . . 573
GetPosition . . . 574
GetRegionOfInterest . . . 577
GetTemperatures . . . 578
GetWeatherData . . . 579
GetWFSPhotometryOptions . . . 580
globals . . . 581
GoNextFlags . . . 586
tcs_photometry::GuideRect . . . 594
tcs_photometry::GuiderMetrologyData . . . 596
hetdex-dither::GuiderRelay . . . 597
guiPreferences . . . 599
guiSound . . . 600
handler_t . . . 602
headerChunk . . . 604
HelloFunctor . . . 605
hetdex-dither::HetdexDitherError . . . 607
HETillumination . . . 610
hex_fit . . . 612
Home . . . 613
HotPixel_t . . . 616
ICSQuery . . . 619
tthread::thread::id . . . 620
IglooRegAlertList . . . 622
IglooStatusField . . . 628

IglooStatusFieldList . . . 629
iglooStatusFieldWidgetSet . . . 630
ExposureInfo::Image . . . 633
Image . . . 634
pasImagePipelineSubSystem::ImageHandlerData . . . 637
ImagePipelineStatus . . . 657
imageRect< Type > . . . 659
init_t . . . 661
initTMCS . . . 662
fixed_point::fixed_internal::ioflags . . . 667
IssuePMACCmd . . . 668
itf_tsf . . . 670
tcsdb::key_descriptor . . . 672
pytcs::tcsdb::key_descriptor . . . 673
legacy_ForceStatusEvents . . . 674
linFit . . . 697
tthread::lock_guard< T > . . . 702
Logger . . . 704
LoggerDB . . . 707
HETillumination::M1motion . . . 711
HETillumination::M4motion . . . 712
m7302::M7302 . . . 713
TCSConfig::mapEntry . . . 718
mbintf_bitfield_t . . . 720
mbintf_hldreg_bitfield_t . . . 721
mbintf_inpreg_analog_t . . . 722
mbintf_inpreg_bitfield_t . . . 724
mbreg_ptr_t . . . 725
mbreg_t . . . 726
mbregid_t . . . 728
MBRegVal . . . 729
md5::MD5 . . . 730
MessageHandlerDescriptor . . . 736
MessageHandler . . . 734
ArchonHandler . . . 257
BusyHandler . . . 270
camra_handlers::CAMRAHandler . . . 303
ConfigHandler . . . 338
GenerateExceptionHandler . . . 532
HelpHandler . . . 606
IglooSensorsHandler . . . 627
InterruptHandler . . . 666
LogRelayDriver . . . 709
MetrologyHandler . . . 742
MonitorDriver . . . 757
namedHandler . . . 785
NOOPHandler . . . 791
P . . . 812
PFIPClientInstrumentHandler . . . 851
PFIPHandler . . . 861
pingHandler . . . 885
PlotterDriver . . . 902
ProxyHandler . . . 970

ServerHandler . . . 1027
ForceStatusEventsHandlerClass . . . 527
ForceStatusEventsHandlerClass . . . 527
SampleHandlerClass . . . 1018
StatusHandler . . . 1162
SubSystemHandler . . . 1194
DevicePipelineEnableCCASDMIHandlerClass . . . 369
DevicePipelineEnableSACDMIHandlerClass . . . 370
DevicePipelineSetGROCHandlerClass . . . 371
DevicePipelineStatusHandlerClass . . . 372
DomeStatusHandlerClass . . . 406
ImagePipelineGetExposureTimeHandlerClass . . . 639
ImagePipelineGetGainHandlerClass . . . 640
ImagePipelineGetOverScanHandlerClass . . . 641
ImagePipelineGetPhotPhotometryOptionsHandlerClass . . . 642
ImagePipelineGetRegionOfInterestHandlerClass . . . 643
ImagePipelineGetWFSPhotometryOptionsHandlerClass . . . 644
ImagePipelineSetCleanImagesHandlerClass . . . 645
ImagePipelineSetCleaningBiasTypeHandlerClass . . . 646
ImagePipelineSetFilterExposureTimeFactorHandlerClass . . . 647
ImagePipelineSetFilterSequenceHandlerClass . . . 648
ImagePipelineSetFilterZeroPointHandlerClass . . . 649
ImagePipelineSetObjectAndMagnitudesHandlerClass . . . 650
ImagePipelineSetOverScanHandlerClass . . . 651
ImagePipelineSetPhotometryOptionsHandlerClass . . . 652
ImagePipelineSetPhotPhotometryOptionsHandlerClass . . . 653
ImagePipelineSetProcessingTypeHandlerClass . . . 654
ImagePipelineSetWFSPhotometryOptionsHandlerClass . . . 655
ImagePipelineStartSendingImagesHandlerClass . . . 656
PipelineCyclePipelineHandlerClass . . . 888
PipelineRelinquishControlHandlerClass . . . 890
PipelineSetCycleTimeHandlerClass . . . 891
PipelineSetIntervalSpeedHandlerClass . . . 892
PipelineStartPipelineHandlerClass . . . 893
PipelineTakeControlHandlerClass . . . 895
ShutterSetPowerHandlerClass . . . 1105
ShutterStatusHandlerClass . . . 1107
StructureMoveToCCASHandlerClass . . . 1179
StructureMoveToStructAzHandlerClass . . . 1180
StructureMoveToTrueAzHandlerClass . . . 1181
StructureOnLineHandlerClass . . . 1182
StructureStatusHandlerClass . . . 1184
StructureStowHandlerClass . . . 1185
WeatherGetDataHandlerClass . . . 1451
WeatherOnLineHandlerClass . . . 1453
TapHandler . . . 1202
TCSHandler . . . 1239
TimeoutHandler . . . 1272
TrackerHandler . . . 1298
UnhandledHandler . . . 1409
UptimeHandler . . . 1414
metadata_entry . . . 739
MetaPayload . . . 740

EventHeader . . . 443
Event . . . 437
MessageHeader . . . 737
Message . . . 731
MirrorDistance . . . 746
AcqMiscStatus::misc_status . . . 747
modbus_mapping_t . . . 751
MODBUSTime . . . 754
mountModel . . . 763
MoveProbeParms . . . 765
moveRTF . . . 769
MoveTo . . . 772
MoveToStructAz . . . 773
moveTSF . . . 774
MsgAttribute . . . 777
tthread::mutex . . . 778
liborchestrate::Named . . . 781
util::NamedReader . . . 786
native_wrapper< Type > . . . 788
native_wrapper< double > . . . 788
index_time . . . 660
julian_day . . . 671
modified_julian_day . . . 755
NoiseAreaData . . . 789
noncopyable . . . 790
CAMRAConfig . . . 298
CAMRAHardware . . . 304
ControllerConfig . . . 348
EPICSRelayConfig . . . 434
EPICSRelayDriver . . . 435
HardwareElement . . . 603
IglooSensors . . . 623
IglooSensorsConfig . . . 624
Message . . . 731
Payload::Data . . . 357
Pool< T > . . . 916
ProcessVariable< T > . . . 961
Proxy . . . 967
ProxyConfig . . . 968
Socket . . . 1113
Client . . . 323
Server . . . 1026
SpectrographConfig . . . 1121
TCSDriver . . . 1218
TCSServerConfig . . . 1244
threadsafe_deque< Entry > . . . 1265
threadsafe_list< Entry > . . . 1266
Trajectory . . . 1308
tthread::ThreadPool . . . 1262
WriteOneWireConfig . . . 1472
tcs_photometry::Object . . . 792
wfs_photometry::WFSCentroid::object_t . . . 794
OnLine . . . 805

Open . . . 806
open_shutter . . . 808
override_ifu_associations . . . 811
ParmsIterator . . . 813
ParmsObject . . . 814
  Correction . . . 351
  itf_coordinate . . . 669
    ShotPoint . . . 1090
    ShotPoint . . . 1090
  pointing_coordinate . . . 912
  ProcessVariable< T > . . . 961
  rtf_coordinate . . . 1016
  TrackerPosition . . . 1300
  Trajectory . . . 1308
  tsf_coordinate . . . 1313
Payload . . . 835
  Event . . . 437
  Message . . . 731
PduOnLine . . . 839
pfip_coord . . . 842
PFIP_EPICSVariable . . . 844
pasImagePipelineSubSystem::PFIPData . . . 857
tcs_photometry::PhotData . . . 872
tcs_photometry::Photometry . . . 873
PIDLoop< M, distance > . . . 883
PIDLoop< double > . . . 883
  MetrologyPIDLoop< double > . . . 744
    DMILoop . . . 383
PIDLoop< T > . . . 883
  MetrologyPIDLoop< T > . . . 744
PIDLoop< vec2_t > . . . 883
  MetrologyPIDLoop< vec2_t > . . . 744
    GuideLoop . . . 593
    GuiderHandler . . . 595
PipelineOnLine . . . 889
tcs_photometry::Pixel . . . 896
Pixel . . . 897
Pixel_t . . . 898
ExposureInfo::PlateScale . . . 899
Point . . . 905
  Trajectory . . . 1308
point . . . 910
Point_t . . . 911
pointing_ForceStatusEvents . . . 913
populate_onewire_device_strings . . . 919
pos_fit . . . 920
fixed_point::fixed_internal::power10< N > . . . 921
fixed_point::fixed_internal::power10< 0 > . . . 922
fixed_point::fixed_internal::power10_aux< N, Even > . . . 923
fixed_point::fixed_internal::power10_aux< N, true > . . . 924
probeAssemblyScene . . . 944
probeData . . . 949

ProbeItem . . . 953
proxy . . . 966
QBlinkingLineEdit . . . 973
QBlinkingLineEditPlugin . . . 974
QGraphicsCircleItem . . . 975
QLedIndicator . . . 976
QLedIndicatorPlugin . . . 978
QSparkLine . . . 979
QSparkLinePlugin . . . 980
QTcsFrame . . . 981
QTcsFramePlugin . . . 982
QTcsProbeAssembly . . . 983
quadFit . . . 985
Quaternion< real > . . . 986
tthread::ratio< N, D > . . . 988
read_now . . . 990
Receiver . . . 992
    ICSDaemon< S > . . . 618
tthread::recursive_mutex . . . 994
wfs_photometry::WFSCentroid::refList_t . . . 996
RelinquishControl . . . 997
Reset . . . 998
reset_clock_voltages . . . 1002
reset_mux . . . 1006
reset_tcs . . . 1010
ExposureInfo::ROI . . . 1014
RowDispatcher . . . 1015
pasDevicePipelineSubSystem::SAMSData . . . 1020
Sender . . . 1022
    DomeClient . . . 403
    PASCameraClient . . . 815
    PFIPClient . . . 850
    PFIPClient . . . 850
    StructureClient . . . 1176
    TCSClient . . . 1206
    TCSClient . . . 1206
    TrackerClient . . . 1293
    WeatherClient . . . 1446
    WeatherClient . . . 1446
ExposureInfo::Sensor . . . 1024
ser7_table_entry . . . 1025
set_bias_offset . . . 1031
set_ccd_temp . . . 1033
set_ccd_temp_xfer . . . 1035
set_clock_voltages . . . 1037
set_cryo_temp_xfer . . . 1038
SetAnalysisRegion . . . 1051
SetBinning . . . 1052
setBufferSize . . . 1053
SetCleanImages . . . 1055
SetCleaningBiasType . . . 1056
SetCycleTime . . . 1057
SetDirection . . . 1058

SetExposuresPerCycle . . . 1060
SetExposureTime . . . 1061
SetFilterExposureTimeFactor . . . 1062
SetFilterSequence . . . 1063
SetFilterZeroPoint . . . 1064
SetGROC . . . 1065
SetImageType . . . 1067
SetIntervalSpeed . . . 1068
SetLightSourcePower . . . 1069
SetLightSourcePowerHoldTime . . . 1070
SetLightSourcePowerLevel . . . 1071
SetNumFlushes . . . 1072
SetObjectAndMagnitudes . . . 1073
SetOverScan . . . 1074
SetPhotPhotometryOptions . . . 1075
SetPixelFormat . . . 1076
SetPower . . . 1077
SetProcessImages . . . 1079
SetProcessingType . . . 1080
SetReadoutSpeed . . . 1081
SetRegionOfInterest . . . 1082
SetStoreImages . . . 1083
SetTargetTemperature . . . 1084
setTraj . . . 1085
SetVelocity . . . 1087
SetWFSPhotometryOptions . . . 1088
shutter_Abort . . . 1093
shutter_Close . . . 1094
shutter_IssuePMACCmd . . . 1095
shutter_OnLine . . . 1096
shutter_Open . . . 1097
shutter_Reset . . . 1098
shutter_SetPower . . . 1099
shutter_Status . . . 1100
AcqShutterStatus::shutter_status . . . 1101
shutter_Stop . . . 1102
sockcmd_reg_id_t . . . 1111
sockcmd_t . . . 1112
SocketException . . . 1114
tcsGui::soundObject . . . 1115
tcsdb::source_descriptor . . . 1116
pytcs::tcsdb::source_descriptor . . . 1117
hetlib::spectrograph::Spectrograph . . . 1118
hetlib::spectrograph::SpectrographList . . . 1123
HETillumination::Spidermotion . . . 1124
fixed_point::fixed_internal::square< N > . . . 1125
StartPipeline . . . 1127
StartSendingImages . . . 1128
StaticConfig . . . 1129
legacyWeatherSubSystem::statistic . . . 1131
WeatherData::statistic . . . 1132
PipelineStatus::status . . . 1133
DomeStatus::status . . . 1134
Status . . . 1135

WeatherData::status . . . 1136
SAMSData::status . . . 1137
legacyWeatherSubSystem::status . . . 1138
PFIPStatusBase::status . . . 1139
APCData::status . . . 1140
TrackerStatus::status . . . 1141
OCDStatus::status . . . 1147
ThermoCubeStatus::status . . . 1148
ShutterStatus::status . . . 1150
TrackerPosition::status . . . 1151
legacyShutterSubSystem::status . . . 1152
VirusMonitorStatus::status . . . 1153
PFIPMonitorStatus::status . . . 1154
DMIData::status . . . 1155
StructureStatus::status . . . 1156
legacyDomeSubSystem::status . . . 1157
IglooData::status . . . 1158
legacyStructureSubSystem::status . . . 1159
StatusBlock< T > . . . 1160
stencil< T > . . . 1163
Stop . . . 1164
StopPipeline . . . 1166
StopSendingImages . . . 1167
Stow . . . 1168
structure_MoveToCCAS . . . 1169
structure_MoveToStructAz . . . 1170
structure_MoveToTrueAz . . . 1171
structure_OnLine . . . 1172
structure_Status . . . 1173
structure_Stop . . . 1174
structure_Stow . . . 1175
StructureModel . . . 1178
subSystem . . . 1190
tcsEvents . . . 1224
liborchestrate::Subsystem . . . 1191
liborchestrate::DmiSubsystem . . . 384
liborchestrate::GuiSubsystem . . . 601
liborchestrate::LegacyDomeSimulator . . . 676
liborchestrate::LegacyShutterSimulator . . . 682
liborchestrate::LegacyStructureSimulator . . . 685
liborchestrate::LegacySubsystem . . . 689
liborchestrate::MonitorSubsystem . . . 758
liborchestrate::PasSubsystem . . . 833
liborchestrate::PfipSubsystem . . . 871
liborchestrate::PlotterTcsSubsystem . . . 903
liborchestrate::PlotterTrackerSubsystem . . . 904
liborchestrate::TcsSubsystem . . . 1251
liborchestrate::TmcsSubsystem . . . 1279
liborchestrate::TrackerSubsystem . . . 1304
liborchestrate::SubsystemManager . . . 1195
pytcs::tcsdb::system_descriptor . . . 1196
tcsdb::system_descriptor . . . 1197
apcTabUi::tabData . . . 1198

TakeControl . . . 1199
TCSApp< Subsys, Config > . . . 1203
TCSBaseException . . . 1205
APCException . . . 247
CAMRAException . . . 302
CoordinateTransformException . . . 349
EPICSException . . . 432
GUIException . . . 598
HETEarthOrientationException . . . 609
IglooSensorsException . . . 626
LegacyException . . . 680
LogClientException . . . 703
LogServerException . . . 710
MessagingException . . . 738
MountModelException . . . 764
NamedException . . . 784
PASException . . . 819
PFIPException . . . 860
PFIPModbusException . . . 864
ProxyException . . . 969
TCSConfigException . . . 1213
TCSException . . . 1225
TrackerException . . . 1297
TrajectoryException . . . 1309
UnknownException . . . 1411
WFSException . . . 1459
TCSConfig . . . 1208
apcCmdConfig . . . 244
apcConfig . . . 245
CAMRAConfig . . . 298
cmdConfig . . . 328
Config . . . 337
consoleConfig . . . 347
ControllerConfig . . . 348
EPICSRelayConfig . . . 434
eventsConfig . . . 448
GLTrackerConfig . . . 582
GroupConfig . . . 587
guiConfig . . . 588
guiConfig . . . 588
IglooSensorsConfig . . . 624
legacyConfig . . . 675
LogRelayConfig . . . 708
MonitorConfig . . . 756
namedConfig . . . 783
pasConfig . . . 816
PFIPConfig . . . 852
PlotterConfig . . . 901
pointingConfig . . . 914
ProxyConfig . . . 968
sampleConfig . . . 1017
SpectrographConfig . . . 1121
SubSystemConfig . . . 1192

CameraConfig . . . 275
CleanerConfig . . . 318
DeviceConfig . . . 368
DomeConfig . . . 404
LightSourceConfig . . . 696
pduConfig . . . 838
PipelineConfig . . . 886
ProcessorConfig . . . 959
ShutterConfig . . . 1103
StructureConfig . . . 1177
WeatherConfig . . . 1447
TCSServerConfig . . . 1244
TopConfig . . . 1282
TrackerConfig . . . 1294
WriteOneWireConfig . . . 1472
pasImagePipelineSubSystem::TCSData . . . 1215
tcsdb::tcsdb . . . 1216
pytcs::tcsdb::tcsdb . . . 1217
TCSEvent::TCSEvent . . . 1221
tcs_event::TCSEvent . . . 1222
pytcs::tcs_event::TCSEvent . . . 1223
TcsException . . . 1226
TCSLog::TCSLog . . . 1240
pytcs::TCSLog::TCSLog . . . 1241
pytcs::TCSNamed::TCSNamed . . . 1242
TCSNamed::TCSNamed . . . 1243
TCSSocket . . . 1245
TCSTelnet . . . 1252
Cntl . . . 329
SubSystem . . . 1188
pytcs::tcssubsystem::TCSSubSystem . . . 1248
tcssubsystem::TCSSubSystem . . . 1250
telescope . . . 1253
TCSClient::TelescopeData . . . 1254
Test . . . 1255
tthread::thread . . . 1258
tthread::ThreadWorker . . . 1268
Config . . . 337
tthread::ThreadPoolException . . . 1263
tthread::ThreadPoolWorker . . . 1264
threadsafe_queue< Entry > . . . 1267
tthread::Timer . . . 1274
timeToIntercept . . . 1276
tle_t . . . 1277
tracCoord . . . 1284
tracker_coordinate . . . 1286
itf_coordinate . . . 669
rtf_coordinate . . . 1016
GLTrackerDriver::tracker_position . . . 1292
TrackerData . . . 1295
HETillumination::Trackermotion . . . 1299
traj_class::Trajectory . . . 1307

traj_class::TrajectoryList . . . 1310
tcs_photometry::TTCAMMetrologyData . . . 1314
turn_off_leds . . . 1316
Ui_aboutDialog . . . 1318
Ui::aboutDialog . . . 209
aboutDialog . . . 210
aboutDialog . . . 210
Ui_acqCamStatusTab . . . 1319
Ui::acqCamStatusTab . . . 224
acqCamStatusTabUi . . . 225
Ui_alertsDialog . . . 1321
Ui::alertsDialog . . . 236
alertsDialog . . . 237
Ui_apcTab . . . 1322
Ui::apcTab . . . 251
apcTabUi . . . 252
Ui_cameraControlTab . . . 1323
Ui::cameraControlTab . . . 278
cameraControlTabUi . . . 279
Ui_cameraImageTab . . . 1325
Ui::cameraImageTab . . . 286
cameraImageTabUi . . . 287
Ui_cameraMetrologyControlTab . . . 1327
Ui::cameraMetrologyControlTab . . . 291
cameraMetrologyControlTabUi . . . 292
Ui_cameraMetrologyDataTab . . . 1328
Ui::cameraMetrologyDataTab . . . 293
cameraMetrologyDataTabUi . . . 294
Ui_configureSystemsDialog . . . 1329
Ui::configureSystemsDialog . . . 341
configureSystemsDialog . . . 342
Ui_configureTabsDialog . . . 1330
Ui::configureTabsDialog . . . 344
configureTabsDialog . . . 343
configureTabsDialog . . . 343
Ui_confirmOffsetDialog . . . 1331
Ui::confirmOffsetDialog . . . 346
confirmOffsetDialog . . . 345
Ui_dmiTab . . . 1332
Ui::dmiTab . . . 385
dmiTabUi . . . 386
Ui_domeTab . . . 1334
Ui::domeTab . . . 407
domeTabUi . . . 408
Ui_editPreferencesDialog . . . 1336
Ui::editPreferencesDialog . . . 413
editPreferencesDialog . . . 412
Ui_eventMonitorTab . . . 1338

Ui::eventMonitorTab . . . 445
eventMonitorTabUi . . . 446
Ui_fcuControlTab . . . 1339
Ui::fcuControlTab . . . 478
fcuControlTabUi . . . 479
Ui_fcuStatusTab . . . 1341
Ui::fcuStatusTab . . . 503
fcuStatusTabUi . . . 504
Ui_fiducialDialog . . . 1343
Ui::fiducialDialog . . . 507
fiducialDialog . . . 506
Ui_focalPlaneShutterTab . . . 1344
Ui::focalPlaneShutterTab . . . 521
focalPlaneShutterTabUi . . . 522
Ui_guideBoxDialog . . . 1345
Ui::guideBoxDialog . . . 592
guideBoxDialog . . . 591
Ui_iglooTab . . . 1346
Ui::iglooTab . . . 631
iglooTabUi . . . 632
Ui_legacyTab . . . 1347
Ui::legacyTab . . . 690
legacyTabUi . . . 691
Ui_loadMarkersDialog . . . 1351
Ui::loadMarkersDialog . . . 701
loadMarkersDialog . . . 700
Ui_mainWindow . . . 1352
Ui::mainWindow . . . 717
mainWindow . . . 714
mainWindow . . . 714
Ui_pfipControlTab . . . 1354
Ui::pfipControlTab . . . 855
pfipControlTabUi . . . 856
Ui_pfipEnvironmentTab . . . 1356
Ui::pfipEnvironmentTab . . . 858
pfipEnvironmentTabUi . . . 859
Ui_photometryOptionsDialog . . . 1358
Ui::photometryOptionsDialog . . . 880
photometryOptionsDialog . . . 881
Ui_probeAssemblyTab . . . 1359
Ui::probeAssemblyTab . . . 945
probeAssemblyTabUi . . . 946
Ui_probeControlTab . . . 1360
Ui::probeControlTab . . . 947
probeControlTabUi . . . 948
Ui_probeGainControlTab . . . 1364
Ui::probeGainControlTab . . . 950
probeGainControlTabUi . . . 951

Ui_probeStatusTab . . . 1365
Ui::probeStatusTab . . . 957
probeStatusTabUi . . . 958
Ui_productionTab . . . 1369
Ui::productionTab . . . 963
productionTabUi . . . 964
Ui_shutterTab . . . 1377
Ui::shutterTab . . . 1108
shutterTabUi . . . 1109
Ui_structureTab . . . 1378
Ui::structureTab . . . 1186
structureTabUi . . . 1187
Ui_trackerStatusTab . . . 1379
Ui::trackerStatusTab . . . 1302
trackerStatusTabUi . . . 1303
Ui_trackerTab . . . 1387
Ui::trackerTab . . . 1305
trackerTabUi . . . 1306
Ui_trajectoryTab . . . 1394
Ui::trajectoryTab . . . 1311
trajectoryTabUi . . . 1312
Ui_videoSwitchTab . . . 1398
Ui::videoSwitchTab . . . 1427
videoSwitchTabUi . . . 1428
Ui_virusEnvironmentTab . . . 1399
Ui::virusEnvironmentTab . . . 1432
virusEnvironmentTabUi . . . 1433
Ui_weatherDataTab . . . 1405
Ui::weatherDataTab . . . 1449
weatherDataTabUi . . . 1450
Ui_webBrowserTab . . . 1408
Ui::webBrowserTab . . . 1454
webBrowserTabUi . . . 1455
utc . . . 1415
utc_iso8601 . . . 1416
validPixelFormat . . . 1420
wfs_photometry::WFSCentroid::vector_t . . . 1421
Venc1PrintRegister . . . 1422
Venc1WriteRegister . . . 1423
Venc2PrintRegister . . . 1424
Venc2WriteRegister . . . 1425
virusEnvironmentTabUi::virusTempData . . . 1435
wait_for_readout . . . 1437
wait_for_shutter_close . . . 1440
wait_for_write . . . 1442
weather . . . 1443
weather_GetData . . . 1444
weather_OnLine . . . 1445
WeatherOnLine . . . 1452

webBrowserTabUiThreadWorker 1456
wfs_photometry::WFSCentroid 1457
wfsData 1458
CentroidInfo 313
FitData 512
ImageData 636
ImageMask 638
ModalFitData 749
ModalMatrix 750
ObscGeom 795
OModalMatrix 804
PhotometryData 879
PSFFitData 971
PupilData 972
SHGrid 1089
SHTrackerData 1091
SlopeMatrix 1110
wfs_photometry::WFSFit 1460
wfs_photometry::WFSFitException 1461
wfslib 1463
wfslib_het 1465
wfslib_het 1465
wfsLogger 1468
WFSToTCSLogLevelMap_t 1470
xsky_find 1474

Chapter 5

Class Index

5.1 Class List

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

_modbus . . . 193
_modbus_backend . . . 194
_modbus_rtu . . . 195
_modbus_tcp . . . 196
_modbus_tcp_pi . . . 197
_sft . . . 198
tthread::_thread_start_functor< T > . . . 199
tthread::_thread_start_functor_arg< F, A > . . . 200
tthread::_thread_start_info . . . 201
tthread::_thread_start_member< 14463331 > . . . 202
tthread::_thread_start_member_arg< 14463331, A > . . . 203
Abort (Abort current Dome command (normally movement)) . . . 204
camra_handlers::abort . . . 205
abort . . . 206
abortMove . . . 207
tracker_handlers::abortMove . . . 208
Ui::aboutDialog . . . 209
aboutDialog . . . 210
ACQ_Camera (A class to represent the acquisition camera) . . . 211
ACQ_Shutter (A class to represent the acquisition camera shutter) . . . 213
ACQ_SubSystem (Used to manage the PFIP hardware, including the Probe Assembly, the Acquisition Camera, and the main Shutter. It uses the EPICS protocol to communicate with the hardware) . . . 217
ACQCamInfo . . . 222
AcqCamStatus . . . 223
Ui::acqCamStatusTab . . . 224
acqCamStatusTabUi . . . 225
AcqCWFSStatus . . . 226
AcqENWAStatus . . . 227
AcqMiscStatus . . . 228
AcqShutterStatus . . . 229
activate . . . 230
add_correction . . . 231

AdjustDither . . . 232
Alarm_Monitor . . . 233
VirusMonitorStatus::alarms . . . 234
PFIPMonitorStatus::alarms . . . 235
Ui::alertsDialog . . . 236
alertsDialog . . . 237
altaz_coordinate . . . 238
Amoeba< Func > . . . 239
Amp (Class corresponding to a single CCD output amplifier) . . . 240
AnalysisRegion . . . 242
apcCmd . . . 243
apcCmdConfig . . . 244
apcConfig . . . 245
APCData . . . 246
APCException . . . 247
apcPduSubSystem . . . 248
apcServer . . . 249
Ui::apcTab . . . 251
apcTabUi . . . 252
tcs_photometry::AperturePhotResults . . . 254
apply_offset . . . 255
Archon . . . 256
ArchonHandler . . . 257
array_dbr_int_t . . . 258
array_dbr_long_t . . . 259
ArrayType . . . 260
tcsdb::attr_descriptor . . . 261
pytcs::tcsdb::attr_descriptor . . . 262
averager< Type > . . . 263
AVT . . . 264
BeginRangeProbes . . . 265
bestDistance . . . 266
ExposureInfo::BINNING . . . 267
Bitfield . . . 268
BuildPupilThreadArgs . . . 269
BusyHandler . . . 270
Camera (Base Camera class) . . . 271
AcqCamStatus::camera_status . . . 273
CameraAVT . . . 274
CameraConfig . . . 275
Ui::cameraControlTab . . . 278
cameraControlTabUi . . . 279
CameraFLI::CameraData . . . 280
CameraFLI . . . 281
cameraImageMarker . . . 282
cameraImageScene . . . 283
Ui::cameraImageTab . . . 286
cameraImageTabUi . . . 287
cameraImageTabUiThreadWorker . . . 289
CameraInfo . . . 290
Ui::cameraMetrologyControlTab . . . 291
cameraMetrologyControlTabUi . . . 292
Ui::cameraMetrologyDataTab . . . 293
cameraMetrologyDataTabUi . . . 294

CameraPGR . . . 295
CameraSim . . . 297
CAMRAConfig . . . 298
CAMRADriver (Encapsulates ARC CArcSys class for remote access) . . . 300
CAMRAException . . . 302
camra_handlers::CAMRAHandler (Base class for all remote interfaces) . . . 303
CAMRAHardware (Hardware encapsulation class) . . . 304
cancel_trajectory . . . 307
CBUF< indexType, size, entryType > . . . 308
geHeaters::Ccd . . . 309
CCD (Class corresponding to a single CCD) . . . 310
center . . . 312
CentroidInfo . . . 313
check . . . 314
chunkHeader . . . 315
cJSON . . . 316
cJSON_Hooks . . . 317
CleanerConfig . . . 318
clear_fiducial . . . 319
clear_trajectories . . . 320
clearError . . . 321
tracker_handlers::clearError . . . 322
Client . . . 323
Close (Close the shutter) . . . 324
camra_handlers::close_shutter . . . 325
close_shutter . . . 326
CloseShutter . . . 327
cmdConfig . . . 328
Cntl (Class corresponding the a single controller) . . . 329
coefs_t . . . 331
Command . . . 332
camra_handlers::command_controller . . . 333
command_controller . . . 334
tthread::condition_variable . . . 335
Config . . . 337
ConfigHandler . . . 338
tcs_photometry::ConfigureParams . . . 339
wfs_photometry::ConfigureParams . . . 340
Ui::configureSystemsDialog . . . 341
configureSystemsDialog . . . 342
configureTabsDialog . . . 343
Ui::configureTabsDialog . . . 344
confirmOffsetDialog . . . 345
Ui::confirmOffsetDialog . . . 346
consoleConfig . . . 347
ControllerConfig . . . 348
CoordinateTransformException . . . 349
correct_for_structure . . . 350
Correction . . . 351
cubFit . . . 352
AcqCWFSStatus::cwfs_status . . . 353
CWFSInfo . . . 354
CyclePipeline . . . 355
mainWindow::DashBoardAlerts . . . 356

Payload::Data . . . 357
dataChunk . . . 358
deactivate . . . 359
deep_arg_t . . . 360
delta_time (UT1 definitions (the deltaUT=UT1-UTC structure)) . . . 362
DeployACQCameraMirror . . . 363
DeployCWFSCamera . . . 364
DeployCWFSCameraMirror . . . 365
DeployPVCamera . . . 366
detachedTabWindow . . . 367
DeviceConfig . . . 368
DevicePipelineEnableCCASDMIHandlerClass . . . 369
DevicePipelineEnableSACDMIHandlerClass . . . 370
DevicePipelineSetGROCHandlerClass . . . 371
DevicePipelineStatusHandlerClass . . . 372
camra_handlers::disable_pressure_gauges . . . 373
disable_pressure_gauges . . . 374
disable_rho . . . 375
camra_handlers::disable_shutter . . . 376
disable_shutter . . . 377
DisableProbes . . . 378
DMI_get_status . . . 379
pasDevicePipelineSubSystem::DMIData . . . 380
DMIData . . . 381
DMIHandler . . . 382
DMILoop . . . 383
liborchestrate::DmiSubsystem . . . 384
Ui::dmiTab . . . 385
dmiTabUi . . . 386
Doh . . . 387
dome_Abort . . . 388
dome_GetPosition . . . 389
dome_Home . . . 390
dome_IssuePMACCmd . . . 391
dome_MoveTo . . . 392
dome_MoveToCCAS . . . 393
dome_OnLine . . . 394
dome_Reset . . . 395
dome_SetDirection . . . 396
dome_SetPower . . . 397
dome_SetVelocity . . . 398
dome_Status . . . 399
dome_Stop . . . 400
dome_Stow . . . 401
dome_Test . . . 402
DomeClient . . . 403
DomeConfig . . . 404
DomeStatus . . . 405
DomeStatusHandlerClass . . . 406
Ui::domeTab . . . 407
domeTabUi . . . 408
droppablePlainTextEdit . . . 409
tthread::chrono::duration< _Rep, _Period > . . . 410
DynamicConfig . . . 411

editPreferencesDialog . . . 412
Ui::editPreferencesDialog . . . 413
enable_pressure_gauges . . . 414
camra_handlers::enable_pressure_gauges . . . 415
enable_rho . . . 416
camra_handlers::enable_shutter . . . 417
enable_shutter . . . 418
EnableCCASDMI . . . 419
EnableProbes . . . 420
EnableSACDMI . . . 421
EndRangeProbes . . . 422
Enum . . . 423
AcqENWAStatus::enwa_status . . . 424
ENWAInfo . . . 425
Ephemeris . . . 426
EphList (Contains all the information about a satellite's track) . . . 427
EphRec (Records read from file for in order to compute a non sideral tracks) . . . 428
EPICSComm . . . 429
EPICSException . . . 432
EPICSHandle . . . 433
EPICSRelayConfig . . . 434
EPICSRelayDriver . . . 435
eq_coords . . . 436
Event . . . 437
replay::EventBuffer . . . 439
eventCmd . . . 440
EventHandler . . . 441
EventHeader . . . 443
EventList . . . 444
Ui::eventMonitorTab . . . 445
eventMonitorTabUi . . . 446
eventMonitorTabUiThreadWorker . . . 447
eventsConfig . . . 448
EventSink . . . 449
EventSource . . . 451
EventSourceRoot . . . 452
EventSourceSocket . . . 453
ExportData . . . 454
camra_handlers::expose . . . 455
expose . . . 456
Exposure . . . 457
ExposureDescriptor . . . 458
ExposureInfo . . . 460
tthread::fast_mutex . . . 462
FCU_Lamp (Manages the state of the lamps. There are 5 lamps on each FCU slider. The FCU_-Lamp class is also used for the single LDLS on the FCU) . . . 464
FCU_LDLS (The class representing the LDLS and its state) . . . 465
FCU_Shutter . . . 466
FCU_Slider (A class to represent the FCU sliders. There are two -- slider a and slider b. Each contains five positions and five lamps) . . . 467
FCU_State (A class to hold state information for the FCU. Has status of FCU Head position and lightGuide position) . . . 468
FCUStatus::fcu_status . . . 470

FCU_SubSystem (Used to control the operation of the Facility Calibration Unit hardware, and provides an API to pfipServer for its management. The FCU_SubSystem contains an instance of the MODBUSComm class, which provides communication to the hardware using the MODBUS protocol. In addition to providing operational changes to the FCU, FCU_SubSystem also maintains status of all the hardware and monitors alarms and notifications about changes to the hardware and its environment. The FCU_SubSystem uses a background thread to update these status changes) 472
FCUCancelExposure 475
FCUCommException 476
FCUCommunication 477
Ui::fcuControlTab 478
fcuControlTabUi 479
FCUDeployHead 480
FCUDisableAllLights 481
FCUEnableLamps 482
FCUEnableLightSources 483
FCUException 484
FCUExposeLDLS 485
FCUExposeSliderA 486
FCUExposeSliderB 487
FCUGetCurrentExposureInfo 488
FCUGetLastExposureInfo 489
FCUHandler 490
FCUInitLightSources 492
FCUMoveLightBank 493
FCUMoveLightGuide 494
FCUMoveSlider 495
FCUMutexException 496
FCUPrintRegister 497
FCURetractHead 498
FCUSetLampWarmUp 499
FCUSetLDLSWarmUp 500
FCUShutDown 501
FCUStatus 502
Ui::fcuStatusTab 503
fcuStatusTabUi 504
FCUWriteRegister 505
fiducialDialog 506
Ui::fiducialDialog 507
pasImagePipelineSubSystem::TCSData::fiducialObject 508
ExposureInfo::FILTER_INFO 509
AcqCamStatus::filter_status 510
Filters 511
FitData 512
tcs_photometry::FitPars 513
pasImagePipelineSubSystem::FITSMapEntry 514
FitsOutWrapper 515
fixed_point::FixedPoint< T, N > 516
PFIPStatusBase::flags 518
flagValues 519
focalPlaneShutterScene 520
Ui::focalPlaneShutterTab 521
focalPlaneShutterTabUi 522
Foo 523

FooBar . . . 524
force_state_event . . . 525
ForceStatusEvent . . . 526
ForceStatusEventsHandlerClass . . . 527
formatChunk . . . 528
fptime . . . 529
fw102b::FW102B . . . 531
GenerateExceptionHandler . . . 532
get . . . 533
get_analysis_region . . . 534
camra_handlers::get_ccd_temp (Read the temperature of the given CCD and replies with the result) . . . 535
get_ccd_temp . . . 536
camra_handlers::get_clock_voltages . . . 537
get_clock_voltages . . . 538
get_correction . . . 539
camra_handlers::get_cryo_pressure . . . 540
get_cryo_pressure . . . 541
camra_handlers::get_cryo_temp . . . 542
get_cryo_temp . . . 543
get_current_local_sidereal_time . . . 544
get_current_pointing . . . 545
get_current_trajectory_info . . . 546
camra_handlers::get_exposure_info . . . 547
get_exposure_info . . . 548
get_fiducial . . . 549
get_focus_gain . . . 550
get_focus_offset . . . 551
get_focus_window_length . . . 552
get_hardware_status . . . 553
camra_handlers::get_hardware_status . . . 554
get_heater_voltage . . . 555
camra_handlers::get_heater_voltage . . . 556
get_itf_offset . . . 557
get_metadata . . . 558
get_next_trajectory_info . . . 559
get_radec_offset . . . 560
get_sky_offset . . . 561
get_status . . . 562
get_tiptilt_gain . . . 563
get_tiptilt_offset . . . 564
get_tiptilt_window_length . . . 565
get_tuning< T > . . . 566
GetACQSystemInfo . . . 567
GetAnalysisRegion . . . 568
GetExposureTime . . . 569
GetGain . . . 570
GetLightSourcePowerLevel . . . 571
GetOverScan . . . 572
GetPhotPhotometryOptions . . . 573
GetPosition (Get current dome position) . . . 574
GetProbeInfo . . . 575
GetProbePosition . . . 576
GetRegionOfInterest . . . 577

GetTemperatures (Get temperatures) . . . 578
GetWeatherData (Return legacy Weather data items) . . . 579
GetWFSPhotometryOptions . . . 580
globals . . . 581
GLTrackerConfig . . . 582
GLTrackerDriver . . . 583
go_next . . . 585
GoNextFlags . . . 586
GroupConfig . . . 587
guiConfig . . . 588
guideBoxDialog . . . 591
Ui::guideBoxDialog . . . 592
GuideLoop . . . 593
tcs_photometry::GuideRect . . . 594
GuiderHandler . . . 595
tcs_photometry::GuiderMetrologyData . . . 596
hetdex-dither::GuiderRelay . . . 597
GUIException . . . 598
guiPreferences . . . 599
guiSound . . . 600
liborchestrate::GuiSubsystem . . . 601
handler_t . . . 602
HardwareElement . . . 603
headerChunk . . . 604
HelloFunctor . . . 605
HelpHandler . . . 606
hetdex-dither::HetdexDitherError . . . 607
HETEarthOrientation . . . 608
HETEarthOrientationException . . . 609
HETillumination . . . 610
hex_fit . . . 612
Home (Command the dome to find its encoder home postion) . . . 613
HomeProbes . . . 614
HomeShutter . . . 615
HotPixel_t . . . 616
ICSConfig . . . 617
ICSDaemon< S > . . . 618
ICSQuery . . . 619
tthread::thread::id . . . 620
IglooData (A class representing the main control values manipulated by the user on the Control tab. Also contains the alert list, for updating the dashboard alert system) . . . 621
IglooRegAlertList (A collection of alert messages for the TCS GUI dashboard alert system) . . . 622
IglooSensors . . . 623
IglooSensorsConfig . . . 624
IglooSensorsConnection . . . 625
IglooSensorsException . . . 626
IglooSensorsHandler . . . 627
IglooStatusField (A structure containing the data for a "field" represented by a row of widgets in the Status tab) . . . 628
IglooStatusFieldList (A class wrapping a map of IglooStatusField instances, storing data for a dynamic list of widgets) . . . 629
iglooStatusFieldWidgetSet (A class representing a row of dynamic widgets pertaining to a particular "field" in the igloo_sensors subsystem interface. Each instance provides the GUI representation of an instance of IglooStatusField) . . . 630

Ui::iglooTab . . . 631
iglooTabUi (Class representing the Igloo tab for TCS GUI) . . . 632
ExposureInfo::Image . . . 633
Image (Simple 2D BGR image class) . . . 634
ImageCleaner . . . 635
ImageData . . . 636
pasImagePipelineSubSystem::ImageHandlerData . . . 637
ImageMask . . . 638
ImagePipelineGetExposureTimeHandlerClass . . . 639
ImagePipelineGetGainHandlerClass . . . 640
ImagePipelineGetOverScanHandlerClass . . . 641
ImagePipelineGetPhotPhotometryOptionsHandlerClass . . . 642
ImagePipelineGetRegionOfInterestHandlerClass . . . 643
ImagePipelineGetWFSPhotometryOptionsHandlerClass . . . 644
ImagePipelineSetCleanImagesHandlerClass . . . 645
ImagePipelineSetCleaningBiasTypeHandlerClass . . . 646
ImagePipelineSetFilterExposureTimeFactorHandlerClass . . . 647
ImagePipelineSetFilterSequenceHandlerClass . . . 648
ImagePipelineSetFilterZeroPointHandlerClass . . . 649
ImagePipelineSetObjectAndMagnitudesHandlerClass . . . 650
ImagePipelineSetOverScanHandlerClass . . . 651
ImagePipelineSetPhotometryOptionsHandlerClass . . . 652
ImagePipelineSetPhotPhotometryOptionsHandlerClass . . . 653
ImagePipelineSetProcessingTypeHandlerClass . . . 654
ImagePipelineSetWFSPhotometryOptionsHandlerClass . . . 655
ImagePipelineStartSendingImagesHandlerClass . . . 656
ImagePipelineStatus . . . 657
ImageProcessor (Base Image Processor class) . . . 658
imageRect< Type > . . . 659
index_time . . . 660
init_t . . . 661
initTMCS . . . 662
tracker_handlers::initTMCS . . . 663
inject_metrology . . . 664
Instrument . . . 665
InterruptHandler . . . 666
fixed_point::fixed_internal::ioflags . . . 667
IssuePMACCmd (Yyy) . . . 668
itf_coordinate . . . 669
itf_tsf . . . 670
julian_day . . . 671
tcsdb::key_descriptor . . . 672
pytcs::tcsdb::key_descriptor . . . 673
legacy_ForceStatusEvents . . . 674
legacyConfig . . . 675
liborchestrate::LegacyDomeSimulator . . . 676
legacyDomeSubSystem (Provides access to the legacy Dome server) . . . 677
LegacyException . . . 680
legacyServer . . . 681
liborchestrate::LegacyShutterSimulator . . . 682
legacyShutterSubSystem (Provides access to the legacy Shutter server) . . . 683
liborchestrate::LegacyStructureSimulator . . . 685
legacyStructureSubSystem (Provides access to the legacy Structure server) . . . 686
liborchestrate::LegacySubsystem . . . 689

Ui::legacyTab . . . 690
legacyTabUi . . . 691
legacyWeatherSubSystem (Provides access to the legacy HET Weather server) . . . 692
LightSource (Base Light Source class) . . . 695
LightSourceConfig . . . 696
linFit . . . 697
load_trajectory . . . 698
loadedUi . . . 699
loadMarkersDialog . . . 700
Ui::loadMarkersDialog . . . 701
tthread::lock_guard< T > . . . 702
LogClientException . . . 703
Logger . . . 704
LoggerDB . . . 707
LogRelayConfig . . . 708
LogRelayDriver . . . 709
LogServerException . . . 710
HETillumination::M1motion . . . 711
HETillumination::M4motion . . . 712
m7302::M7302 . . . 713
mainWindow . . . 714
Ui::mainWindow . . . 717
TCSConfig::mapEntry . . . 718
MAXONStatusInfo . . . 719
mbintf_bitfield_t . . . 720
mbintf_hldreg_bitfield_t . . . 721
mbintf_inpreg_analog_t . . . 722
mbintf_inpreg_bitfield_t . . . 724
mbreg_ptr_t . . . 725
mbreg_t . . . 726
mbregid_t . . . 728
MBRegVal (Message handling abstraction providing get/set access to register values, via the modbus communication thread IglooSensorsConnection::comm_thread()) . . . 729
md5::MD5 . . . 730
Message . . . 731
MessageHandler . . . 734
MessageHandlerDescriptor . . . 736
MessageHeader . . . 737
MessagingException . . . 738
metadata_entry . . . 739
MetaPayload . . . 740
Metrology . . . 741
MetrologyHandler . . . 742
MetrologyLoop . . . 743
MetrologyPIDLoop< T > . . . 744
MidRangeProbes . . . 745
MirrorDistance . . . 746
AcqMiscStatus::misc_status . . . 747
MiscInfo . . . 748
ModalFitData . . . 749
ModalMatrix . . . 750
modbus_mapping_t . . . 751
MODBUSComm . . . 752
MODBUSTime . . . 754

modified_julian_day . . . 755
MonitorConfig . . . 756
MonitorDriver . . . 757
liborchestrate::MonitorSubsystem . . . 758
MonPrintRegister . . . 759
MonWriteRegister . . . 760
MountModel . . . 761
mountModel . . . 763
MountModelException . . . 764
MoveProbeParms . . . 765
MoveProbes . . . 766
MoveProbes_Enc . . . 767
MoveProbes_RaDec . . . 768
moveRTF . . . 769
tracker_handlers::moveRTF . . . 771
MoveTo (Move the dome to the specified azimuth) . . . 772
MoveToStructAz (Move the structure to the specified azimuth without mount model applied) . . . 773
moveTSF . . . 774
tracker_handlers::moveTSF . . . 776
MsgAttribute . . . 777
tthread::mutex . . . 778
Mux (Class corresponding to the controller multiplexing hardware) . . . 780
liborchestrate::Named . . . 781
namedClient . . . 782
namedConfig . . . 783
NamedException . . . 784
namedHandler . . . 785
util::NamedReader . . . 786
namedServer . . . 787
native_wrapper< Type > . . . 788
NoiseAreaData . . . 789
noncopyable . . . 790
NOOPHandler . . . 791
tcs_photometry::Object . . . 792
wfs_photometry::WFSCentroid::object_t . . . 794
ObscGeom . . . 795
OCDStatus . . . 796
offset_fiducial . . . 797
offset_focus . . . 798
offset_itf . . . 799
offset_probe . . . 800
offset_sky . . . 801
offset_tiptilt . . . 802
offset_trajectory . . . 803
OModalMatrix . . . 804
OnLine (Check to see if the legacy dome sub system is connected) . . . 805
Open (Open the shutter) . . . 806
camra_handlers::open_shutter . . . 807
open_shutter . . . 808
OpenShutter . . . 809
camra_handlers::override_ifu_associations . . . 810
override_ifu_associations . . . 811
P . . . 812
ParmsIterator . . . 813

ParmsObject . . . 814
PASCameraClient . . . 815
pasConfig . . . 816
pasDevicePipelineSubSystem . . . 817
PASException . . . 819
pasImageCleanerStandard . . . 820
pasImagePipelineSubSystem . . . 821
pasImageProcessorPhot . . . 826
pasImageProcessorWFS . . . 827
pasImageStorage (Base Image Storage class) . . . 828
pasImageStorageFITS . . . 829
pasPipelineSubSystem . . . 830
pasServer . . . 832
liborchestrate::PasSubsystem . . . 833
pasTTLaserLightSource . . . 834
Payload . . . 835
pduConfig . . . 838
PduOnLine (Check to see if Pdu is running) . . . 839
pfip_clear_positions . . . 840
pfip_close_shutter . . . 841
pfip_coord . . . 842
PFIP_EPICSVariable . . . 844
pfip_insert_acquisition_pickoff . . . 845
PFIP_Monitor . . . 846
pfip_move_probes . . . 847
pfip_open_shutter . . . 848
pfip_withdraw_acquisition_pickoff . . . 849
PFIPClient . . . 850
PFIPClientInstrumentHandler . . . 851
PFIPConfig (A class to handle the PFIP Configuration file and settings, extending the base TCSConfig class) . . . 852
Ui::pfipControlTab . . . 855
pfipControlTabUi . . . 856
pasImagePipelineSubSystem::PFIPData . . . 857
Ui::pfipEnvironmentTab . . . 858
pfipEnvironmentTabUi . . . 859
PFIPException . . . 860
PFIPHandler (A class derived from Handler to encompass all the message handlers for PFIP server) . . . 861
PFIPInterrupt . . . 862
PFIPLampsDB . . . 863
PFIPModbusException . . . 864
PFIPMonitorStatus . . . 865
PFIPProbeInfo (A class encapsulating hardware status for PFIP probes) . . . 866
pfipServer . . . 868
PFIPStatusActions . . . 869
PFIPStatusBase . . . 870
liborchestrate::PfipSubsystem . . . 871
tcs_photometry::PhotData . . . 872
tcs_photometry::Photometry . . . 873
PhotometryData . . . 879
Ui::photometryOptionsDialog . . . 880
photometryOptionsDialog . . . 881
PIDHandler< T > . . . 882

PIDLoop< M, distance > . . . 883
pingHandler . . . 885
PipelineConfig . . . 886
PipelineCyclePipelineHandlerClass . . . 888
PipelineOnLine (Check to see if pipeline is running) . . . 889
PipelineRelinquishControlHandlerClass . . . 890
PipelineSetCycleTimeHandlerClass . . . 891
PipelineSetIntervalSpeedHandlerClass . . . 892
PipelineStartPipelineHandlerClass . . . 893
PipelineStatus . . . 894
PipelineTakeControlHandlerClass . . . 895
tcs_photometry::Pixel . . . 896
Pixel (BGR pixel class) . . . 897
Pixel_t . . . 898
ExposureInfo::PlateScale . . . 899
plotEventsTabUi . . . 900
PlotterConfig . . . 901
PlotterDriver . . . 902
liborchestrate::PlotterTcsSubsystem . . . 903
liborchestrate::PlotterTrackerSubsystem . . . 904
Point . . . 905
point . . . 910
Point_t . . . 911
pointing_coordinate . . . 912
pointing_ForceStatusEvents . . . 913
pointingConfig . . . 914
pointingServer . . . 915
Pool< T > . . . 916
camra_handlers::populate_onewire_device_strings . . . 918
populate_onewire_device_strings . . . 919
pos_fit . . . 920
fixed_point::fixed_internal::power10< N > . . . 921
fixed_point::fixed_internal::power10< 0 > . . . 922
fixed_point::fixed_internal::power10_aux< N, Even > . . . 923
fixed_point::fixed_internal::power10_aux< N, true > . . . 924
PowerACQCamera . . . 925
PowerBIBCamera . . . 926
PowerHPF_ACQCamera . . . 927
PowerHR_SASCamera . . . 928
PowerOffAll . . . 929
PowerOnAll . . . 930
PowerProbeCamera . . . 931
PowerPV_CWFSCamera . . . 932
PowerSBC1 . . . 933
PowerSBC2 . . . 934
PowerSwitch1 . . . 935
PowerSwitch2 . . . 936
PowerTTCamera . . . 937
Probe . . . 938
ProbeAssembly (A class to represent the probe assembly. Handles the coordinated movement and collision avoidance for the four probes) . . . 941
probeAssemblyScene . . . 944
Ui::probeAssemblyTab . . . 945
probeAssemblyTabUi . . . 946

Ui::probeControlTab . . . 947
probeControlTabUi . . . 948
probeData . . . 949
Ui::probeGainControlTab . . . 950
probeGainControlTabUi . . . 951
ProbeInfo . . . 952
ProbeItem . . . 953
ProbePosition . . . 954
ProbeStatus . . . 956
Ui::probeStatusTab . . . 957
probeStatusTabUi . . . 958
ProcessorConfig . . . 959
ProcessVariable< T > . . . 961
Ui::productionTab . . . 963
productionTabUi . . . 964
proxy . . . 966
Proxy . . . 967
ProxyConfig . . . 968
ProxyException . . . 969
ProxyHandler . . . 970
PSFFitData . . . 971
PupilData . . . 972
QBlinkingLineEdit . . . 973
QBlinkingLineEditPlugin . . . 974
QGraphicsCircleItem . . . 975
QLedIndicator . . . 976
QLedIndicatorPlugin . . . 978
QSparkLine . . . 979
QSparkLinePlugin . . . 980
QTcsFrame . . . 981
QTcsFramePlugin . . . 982
QTcsProbeAssembly . . . 983
quadFit . . . 985
Quaternion< real > . . . 986
query_trajectory . . . 987
tthread::ratio< N, D > . . . 988
camra_handlers::read_now . . . 989
read_now . . . 990
ReadRegister . . . 991
Receiver . . . 992
tthread::recursive_mutex . . . 994
wfs_photometry::WFSCentroid::refList_t . . . 996
RelinquishControl . . . 997
Reset (Reset the dome power inverter) . . . 998
camra_handlers::reset . . . 999
reset . . . 1000
camra_handlers::reset_clock_voltages . . . 1001
reset_clock_voltages . . . 1002
reset_focus_offset . . . 1003
reset_itf_offset . . . 1004
camra_handlers::reset_mux . . . 1005
reset_mux . . . 1006
reset_radec_offset . . . 1007
reset_sky_offset . . . 1008

camra_handlers::reset_tcs . . . 1009
reset_tcs . . . 1010
reset_tiptilt_offset . . . 1011
RetractACQCameraMirror . . . 1012
RetractCWFSCameraMirror . . . 1013
ExposureInfo::ROI . . . 1014
RowDispatcher . . . 1015
rtf_coordinate . . . 1016
sampleConfig . . . 1017
SampleHandlerClass . . . 1018
sampleServer . . . 1019
pasDevicePipelineSubSystem::SAMSData . . . 1020
SAMSData . . . 1021
Sender . . . 1022
ExposureInfo::Sensor . . . 1024
ser7_table_entry . . . 1025
Server . . . 1026
ServerHandler . . . 1027
set . . . 1028
set_analysis_region . . . 1029
camra_handlers::set_bias_offset . . . 1030
set_bias_offset . . . 1031
camra_handlers::set_ccd_temp . . . 1032
set_ccd_temp . . . 1033
camra_handlers::set_ccd_temp_xfer . . . 1034
set_ccd_temp_xfer . . . 1035
camra_handlers::set_clock_voltages . . . 1036
set_clock_voltages . . . 1037
set_cryo_temp_xfer . . . 1038
camra_handlers::set_cryo_temp_xfer . . . 1039
set_fiducial . . . 1040
set_filter . . . 1041
set_focus . . . 1042
set_focus_gain . . . 1043
set_focus_window_length . . . 1044
set_metadata . . . 1045
set_position . . . 1046
set_tiptilt_gain . . . 1047
set_tiptilt_window_length . . . 1048
SetACQCameraFilter . . . 1049
SetACQCameraFocus . . . 1050
SetAnalysisRegion . . . 1051
SetBinning (Set camera binning) . . . 1052
setBufferSize . . . 1053
tracker_handlers::setBufferSize . . . 1054
SetCleanImages . . . 1055
SetCleaningBiasType . . . 1056
SetCycleTime . . . 1057
SetDirection (Set the dome movement direction) . . . 1058
SetENWAFilter . . . 1059
SetExposuresPerCycle . . . 1060
SetExposureTime (Set exposure time) . . . 1061
SetFilterExposureTimeFactor . . . 1062
SetFilterSequence . . . 1063

SetFilterZeroPoint . . . 1064
SetGROC . . . 1065
SetGuideCameraFilter . . . 1066
SetImageType (Set image type) . . . 1067
SetIntervalSpeed (Set the pipeline interval speed (time between images)) . . . 1068
SetLightSourcePower . . . 1069
SetLightSourcePowerHoldTime . . . 1070
SetLightSourcePowerLevel . . . 1071
SetNumFlushes (Set number of flushes) . . . 1072
SetObjectAndMagnitudes . . . 1073
SetOverScan . . . 1074
SetPhotPhotometryOptions . . . 1075
SetPixelFormat (Set exposure type) . . . 1076
SetPower (Set the dome power state) . . . 1077
SetProbeGains . . . 1078
SetProcessImages . . . 1079
SetProcessingType . . . 1080
SetReadoutSpeed (Set readout speed) . . . 1081
SetRegionOfInterest (Set sub-image rectangle) . . . 1082
SetStoreImages . . . 1083
SetTargetTemperature (Set target temperature) . . . 1084
setTraj . . . 1085
tracker_handlers::setTraj . . . 1086
SetVelocity (Set the dome movement velocity) . . . 1087
SetWFSPhotometryOptions . . . 1088
SHGrid . . . 1089
ShotPoint (Structure containing tracker variables; see HET TR 43 for details) . . . 1090
SHTrackerData . . . 1091
Shutter . . . 1092
shutter_Abort . . . 1093
shutter_Close . . . 1094
shutter_IssuePMACCmd . . . 1095
shutter_OnLine . . . 1096
shutter_Open . . . 1097
shutter_Reset . . . 1098
shutter_SetPower . . . 1099
shutter_Status . . . 1100
AcqShutterStatus::shutter_status . . . 1101
shutter_Stop . . . 1102
ShutterConfig . . . 1103
ShutterInfo . . . 1104
ShutterSetPowerHandlerClass . . . 1105
ShutterStatus . . . 1106
ShutterStatusHandlerClass . . . 1107
Ui::shutterTab . . . 1108
shutterTabUi . . . 1109
SlopeMatrix . . . 1110
sockcmd_reg_id_t . . . 1111
sockcmd_t . . . 1112
Socket . . . 1113
SocketException . . . 1114
tcsGui::soundObject . . . 1115
tcsdb::source_descriptor . . . 1116
pytcs::tcsdb::source_descriptor . . . 1117

hetlib::spectrograph::Spectrograph . . . 1118
SpectrographConfig (Spectrograph configuration class) . . . 1121
hetlib::spectrograph::SpectrographList . . . 1123
HETillumination::Spidermotion . . . 1124
fixed_point::fixed_internal::square< N > . . . 1125
start . . . 1126
StartPipeline (Start the pipleine for a particular camera) . . . 1127
StartSendingImages (Start sending images to the sender of this message) . . . 1128
StaticConfig . . . 1129
legacyWeatherSubSystem::statistic . . . 1131
WeatherData::statistic . . . 1132
PipelineStatus::status . . . 1133
DomeStatus::status . . . 1134
Status (Return the dome status information) . . . 1135
WeatherData::status . . . 1136
SAMSData::status . . . 1137
legacyWeatherSubSystem::status . . . 1138
PFIPStatusBase::status . . . 1139
APCData::status . . . 1140
TrackerStatus::status . . . 1141
OCDStatus::status . . . 1147
ThermoCubeStatus::status . . . 1148
ShutterStatus::status . . . 1150
TrackerPosition::status . . . 1151
legacyShutterSubSystem::status . . . 1152
VirusMonitorStatus::status . . . 1153
PFIPMonitorStatus::status . . . 1154
DMIData::status . . . 1155
StructureStatus::status . . . 1156
legacyDomeSubSystem::status . . . 1157
IglooData::status . . . 1158
legacyStructureSubSystem::status . . . 1159
StatusBlock< T > . . . 1160
StatusData . . . 1161
StatusHandler . . . 1162
stencil< T > . . . 1163
Stop (Stop the current dome command (normally movement)) . . . 1164
stop . . . 1165
StopPipeline (Stop the pipleine for a particular camera) . . . 1166
StopSendingImages (Stop sending images to the sender of this message) . . . 1167
Stow (Stow the structure) . . . 1168
structure_MoveToCCAS . . . 1169
structure_MoveToStructAz . . . 1170
structure_MoveToTrueAz . . . 1171
structure_OnLine . . . 1172
structure_Status . . . 1173
structure_Stop . . . 1174
structure_Stow . . . 1175
StructureClient . . . 1176
StructureConfig . . . 1177
StructureModel (Legacy Mount Model) . . . 1178
StructureMoveToCCASHandlerClass . . . 1179
StructureMoveToStructAzHandlerClass . . . 1180
StructureMoveToTrueAzHandlerClass . . . 1181

StructureOnLineHandlerClass . . . 1182
StructureStatus . . . 1183
StructureStatusHandlerClass . . . 1184
StructureStowHandlerClass . . . 1185
Ui::structureTab . . . 1186
structureTabUi . . . 1187
SubSystem . . . 1188
subSystem . . . 1190
liborchestrate::Subsystem . . . 1191
SubSystemConfig . . . 1192
SubSystemHandler . . . 1194
liborchestrate::SubsystemManager . . . 1195
pytcs::tcsdb::system_descriptor . . . 1196
tcsdb::system_descriptor . . . 1197
apcTabUi::tabData . . . 1198
TakeControl . . . 1199
TakeExposure . . . 1200
Tap . . . 1201
TapHandler . . . 1202
TCSApp< Subsys, Config > . . . 1203
TCSBaseClass . . . 1204
TCSBaseException . . . 1205
TCSClient . . . 1206
tcsCmd . . . 1207
TCSConfig (Configuration class for HET applcations) . . . 1208
TCSConfigException . . . 1213
tcsConsole . . . 1214
pasImagePipelineSubSystem::TCSData . . . 1215
tcsdb::tcsdb . . . 1216
pytcs::tcsdb::tcsdb . . . 1217
TCSDriver . . . 1218
TCSEvent::TCSEvent . . . 1221
tcs_event::TCSEvent . . . 1222
pytcs::tcs_event::TCSEvent . . . 1223
tcsEvents . . . 1224
TCSException . . . 1225
TcsException . . . 1226
tcsGui . . . 1227
tcsGuiThreadWorker . . . 1233
TCSHandler . . . 1239
TCSLog::TCSLog . . . 1240
pytcs::TCSLog::TCSLog . . . 1241
pytcs::TCSNamed::TCSNamed . . . 1242
TCSNamed::TCSNamed . . . 1243
TCSServerConfig . . . 1244
TCSSocket . . . 1245
pytcs::tcssubsystem::TCSSubSystem . . . 1248
TCSSubSystem . . . 1249
tcssubsystem::TCSSubSystem . . . 1250
liborchestrate::TcsSubsystem . . . 1251
TCSTelnet . . . 1252
telescope (Structure containing information about the HET) . . . 1253
TCSClient::TelescopeData . . . 1254
Test (Start/Stop the dome movement test) . . . 1255

test . . . 1256
ThermoCubeStatus . . . 1257
tthread::thread (Thread class) . . . 1258
tthread::ThreadPool . . . 1262
tthread::ThreadPoolException . . . 1263
tthread::ThreadPoolWorker . . . 1264
threadsafe_deque< Entry > . . . 1265
threadsafe_list< Entry > . . . 1266
threadsafe_queue< Entry > . . . 1267
tthread::ThreadWorker . . . 1268
TimeoutHandler . . . 1272
TimeoutNextMove . . . 1273
tthread::Timer . . . 1274
tracker_handlers::timeToIntercept . . . 1275
timeToIntercept . . . 1276
tle_t . . . 1277
TMCSState . . . 1278
liborchestrate::TmcsSubsystem . . . 1279
toggle . . . 1280
toggle_gui_token . . . 1281
TopConfig . . . 1282
TopDriver . . . 1283
tracCoord (Structure for internal coordinates) . . . 1284
tracker_abort . . . 1285
tracker_coordinate . . . 1286
tracker_init . . . 1287
tracker_move_angular . . . 1288
tracker_move_itf . . . 1289
tracker_move_tsf . . . 1290
tracker_position . . . 1291
GLTrackerDriver::tracker_position . . . 1292
TrackerClient . . . 1293
TrackerConfig . . . 1294
TrackerData . . . 1295
TrackerDriver . . . 1296
TrackerException . . . 1297
TrackerHandler (Base class for all remote interfaces) . . . 1298
HETillumination::Trackermotion . . . 1299
TrackerPosition . . . 1300
TrackerStatus . . . 1301
Ui::trackerStatusTab . . . 1302
trackerStatusTabUi . . . 1303
liborchestrate::TrackerSubsystem . . . 1304
Ui::trackerTab . . . 1305
trackerTabUi . . . 1306
traj_class::Trajectory . . . 1307
Trajectory . . . 1308
TrajectoryException . . . 1309
traj_class::TrajectoryList . . . 1310
Ui::trajectoryTab . . . 1311
trajectoryTabUi . . . 1312
tsf_coordinate . . . 1313
tcs_photometry::TTCAMMetrologyData . . . 1314
tune< T > . . . 1315

turn_off_leds . . . 1316
camra_handlers::turn_off_leds . . . 1317
Ui_aboutDialog . . . 1318
Ui_acqCamStatusTab . . . 1319
Ui_alertsDialog . . . 1321
Ui_apcTab . . . 1322
Ui_cameraControlTab . . . 1323
Ui_cameraImageTab . . . 1325
Ui_cameraMetrologyControlTab . . . 1327
Ui_cameraMetrologyDataTab . . . 1328
Ui_configureSystemsDialog . . . 1329
Ui_configureTabsDialog . . . 1330
Ui_confirmOffsetDialog . . . 1331
Ui_dmiTab . . . 1332
Ui_domeTab . . . 1334
Ui_editPreferencesDialog . . . 1336
Ui_eventMonitorTab . . . 1338
Ui_fcuControlTab . . . 1339
Ui_fcuStatusTab . . . 1341
Ui_fiducialDialog . . . 1343
Ui_focalPlaneShutterTab . . . 1344
Ui_guideBoxDialog . . . 1345
Ui_iglooTab . . . 1346
Ui_legacyTab . . . 1347
Ui_loadMarkersDialog . . . 1351
Ui_mainWindow . . . 1352
Ui_pfipControlTab . . . 1354
Ui_pfipEnvironmentTab . . . 1356
Ui_photometryOptionsDialog . . . 1358
Ui_probeAssemblyTab . . . 1359
Ui_probeControlTab . . . 1360
Ui_probeGainControlTab . . . 1364
Ui_probeStatusTab . . . 1365
Ui_productionTab . . . 1369
Ui_shutterTab . . . 1377
Ui_structureTab . . . 1378
Ui_trackerStatusTab . . . 1379
Ui_trackerTab . . . 1387
Ui_trajectoryTab . . . 1394
Ui_videoSwitchTab . . . 1398
Ui_virusEnvironmentTab . . . 1399
Ui_weatherDataTab . . . 1405
Ui_webBrowserTab . . . 1408
UnhandledHandler . . . 1409
Unimplemented . . . 1410
UnknownException . . . 1411
unset_position . . . 1412
update_setup . . . 1413
UptimeHandler . . . 1414
utc . . . 1415
utc_iso8601 . . . 1416
utilityGui . . . 1417
utilityGuiThreadWorker . . . 1419
validPixelFormat . . . 1420

wfs_photometry::WFSCentroid::vector_t . . . 1421
Venc1PrintRegister . . . 1422
Venc1WriteRegister . . . 1423
Venc2PrintRegister . . . 1424
Venc2WriteRegister . . . 1425
VENCHandler (A class derived from PFIPHandler to provide message handlers for FCU commands) . . . 1426
Ui::videoSwitchTab . . . 1427
videoSwitchTabUi . . . 1428
videoSwitchTabUiThreadWorker . . . 1430
VIRUS_Monitor . . . 1431
Ui::virusEnvironmentTab . . . 1432
virusEnvironmentTabUi . . . 1433
VirusMonitorStatus . . . 1434
virusEnvironmentTabUi::virusTempData . . . 1435
camra_handlers::wait_for_readout . . . 1436
wait_for_readout . . . 1437
wait_for_setup . . . 1438
camra_handlers::wait_for_shutter_close . . . 1439
wait_for_shutter_close . . . 1440
camra_handlers::wait_for_write . . . 1441
wait_for_write . . . 1442
weather . . . 1443
weather_GetData . . . 1444
weather_OnLine . . . 1445
WeatherClient . . . 1446
WeatherConfig . . . 1447
WeatherData . . . 1448
Ui::weatherDataTab . . . 1449
weatherDataTabUi . . . 1450
WeatherGetDataHandlerClass . . . 1451
WeatherOnLine (Check to see if the Weather legacy server is connected) . . . 1452
WeatherOnLineHandlerClass . . . 1453
Ui::webBrowserTab . . . 1454
webBrowserTabUi . . . 1455
webBrowserTabUiThreadWorker . . . 1456
wfs_photometry::WFSCentroid . . . 1457
wfsData . . . 1458
WFSException . . . 1459
wfs_photometry::WFSFit . . . 1460
wfs_photometry::WFSFitException . . . 1461
WFSHandler . . . 1462
wfslib . . . 1463
wfslib_het . . . 1465
wfsLogger . . . 1468
WFSLoop . . . 1469
WFSToTCSLogLevelMap_t . . . 1470
WriteOneWire . . . 1471
WriteOneWireConfig . . . 1472
WriteRegister . . . 1473
xsky_find . . . 1474

Chapter 6

File Index

6.1 File List

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

apc/include/**apcCmd.h** . . . . . **??**
apc/include/**apcCmdConfig.h** . . . . . **??**
apc/include/**apcConfig.h** . . . . . **??**
apc/include/**apcPDUSubSystem.h** . . . . . **??**
apc/include/**apcServer.h** . . . . . **??**
camra/**camra_config.h** . . . . . **??**
camra/**camra_driver.h** . . . . . **??**
camra/**camra_handler_doc.h** . . . . . **??**
camra/**camra_handlers.h** . . . . . **??**
camra/**camra_hardware.h** . . . . . **??**
camra/**camra_server.h** . . . . . **??**
camra/**exposure.h** . . . . . **??**
camra/**md5.h** . . . . . **??**
camra/**pfip_client.h** . . . . . **??**
camra/**shutter.h** . . . . . **??**
camra/**tcs_client.h** . . . . . **??**
camra/**weather_client.h** . . . . . **??**
epics_relay/**array_types.h** . . . . . **??**
epics_relay/**config.h** . . . . . **??**
epics_relay/**driver.h** . . . . . **??**
epics_relay/**epics_handle.h** . . . . . **??**
epics_relay/**epics_relay.h** . . . . . **??**
epics_relay/**process_variable.h** . . . . . **??**
gui/build/**ui_aboutDlg.h** . . . . . **??**
gui/build/**ui_acqCamStatusTab.h** . . . . . **??**
gui/build/**ui_alertsDlg.h** . . . . . **??**
gui/build/**ui_apcTab.h** . . . . . **??**
gui/build/**ui_cameraControlTab.h** . . . . . **??**
gui/build/**ui_cameraImageTab.h** . . . . . **??**
gui/build/**ui_cameraMetrologyControlTab.h** . . . . . **??**
gui/build/**ui_cameraMetrologyDataTab.h** . . . . . **??**
gui/build/**ui_configureSystemsDlg.h** . . . . . **??**
gui/build/**ui_configureTabsDlg.h** . . . . . **??**

gui/build/**ui_confirmOffsetDlg.h** . . . ??
gui/build/**ui_dmiTab.h** . . . ??
gui/build/**ui_domeTab.h** . . . ??
gui/build/**ui_editPreferencesDlg.h** . . . ??
gui/build/**ui_eventMonitorTab.h** . . . ??
gui/build/**ui_fcuControlTab.h** . . . ??
gui/build/**ui_fcuStatusTab.h** . . . ??
gui/build/**ui_fiducialDlg.h** . . . ??
gui/build/**ui_focalPlaneShutterTab.h** . . . ??
gui/build/**ui_guideBoxDlg.h** . . . ??
gui/build/**ui_iglooTab.h** . . . ??
gui/build/**ui_legacyTab.h** . . . ??
gui/build/**ui_loadMarkersDlg.h** . . . ??
gui/build/**ui_mainWindow.h** . . . ??
gui/build/**ui_pfipControlTab.h** . . . ??
gui/build/**ui_pfipEnvironmentTab.h** . . . ??
gui/build/**ui_photometryOptionsDlg.h** . . . ??
gui/build/**ui_probeAssemblyTab.h** . . . ??
gui/build/**ui_probeControlTab.h** . . . ??
gui/build/**ui_probeGainControlTab.h** . . . ??
gui/build/**ui_probeStatusTab.h** . . . ??
gui/build/**ui_productionTab.h** . . . ??
gui/build/**ui_shutterTab.h** . . . ??
gui/build/**ui_structureTab.h** . . . ??
gui/build/**ui_trackerStatusTab.h** . . . ??
gui/build/**ui_trackerTab.h** . . . ??
gui/build/**ui_trajectoryTab.h** . . . ??
gui/build/**ui_videoSwitchTab.h** . . . ??
gui/build/**ui_virusEnvironmentTab.h** . . . ??
gui/build/**ui_weatherDataTab.h** . . . ??
gui/build/**ui_webBrowserTab.h** . . . ??
gui/designer/customWidgetPlugins/blinkingLineEdit/**qBlinkingLineEdit.h** . . . ??
gui/designer/customWidgetPlugins/blinkingLineEdit/**qBlinkingLineEditPlugin.h** . . . ??
gui/designer/customWidgetPlugins/ledIndicator/**qLedIndicator.h** . . . ??
gui/designer/customWidgetPlugins/ledIndicator/**qLedIndicatorPlugin.h** . . . ??
gui/designer/customWidgetPlugins/sparkLine/**qSparkLine.h** . . . ??
gui/designer/customWidgetPlugins/sparkLine/**qSparkLinePlugin.h** . . . ??
gui/designer/customWidgetPlugins/tcsFrame/**qTcsFrame.h** . . . ??
gui/designer/customWidgetPlugins/tcsFrame/**qTcsFramePlugin.h** . . . ??
gui/include/**acqCamStatusTabUi.h** . . . ??
gui/include/**apcStatusData.h** . . . ??
gui/include/**apcTabUi.h** . . . ??
gui/include/**cameraControlTabUi.h** . . . ??
gui/include/**cameraImageScene.h** . . . ??
gui/include/**cameraImageTabUi.h** . . . ??
gui/include/**cameraMetrologyControlTabUi.h** . . . ??
gui/include/**cameraMetrologyDataTabUi.h** . . . ??
gui/include/**configureSystemsDlg.h** . . . ??
gui/include/**configureTabsDlg.h** . . . ??
gui/include/**dmiTabUi.h** . . . ??
gui/include/**domeTabUi.h** . . . ??
gui/include/**drawShapes.h** . . . ??
gui/include/**droppablePlainTextEdit.h** . . . ??
gui/include/**editPreferencesDlg.h** . . . ??

gui/include/**eventMonitorTabUi.h** . . . ??
gui/include/**fcuControlTabUi.h** . . . ??
gui/include/**fcuStatusTabUi.h** . . . ??
gui/include/**focalPlaneShutter.h** . . . ??
gui/include/**focalPlaneShutterTabUi.h** . . . ??
gui/include/**guiConfig.h** . . . ??
gui/include/**guiPreferences.h** . . . ??
gui/include/**guiSound.h** . . . ??
gui/include/**iglooStatusData.h** . . . ??
gui/include/**iglooTabUi.h** . . . ??
gui/include/**legacyStatusData.h** . . . ??
gui/include/**legacyTabUi.h** . . . ??
gui/include/**loadedUi.h** . . . ??
gui/include/**loadMarkersDlg.h** . . . ??
gui/include/**mainWindow.h** . . . ??
gui/include/**ocdStatusData.h** . . . ??
gui/include/**pasStatusData.h** . . . ??
gui/include/**pfipControlTabUi.h** . . . ??
gui/include/**pfipEnvironmentTabUi.h** . . . ??
gui/include/**pfipStatusData.h** . . . ??
gui/include/**photometryOptionsDlg.h** . . . ??
gui/include/**pipelineStatusData.h** . . . ??
gui/include/**probeAssembly.h** . . . ??
gui/include/**probeAssemblyTabUi.h** . . . ??
gui/include/**probeControlTabUi.h** . . . ??
gui/include/**probeGainControlTabUi.h** . . . ??
gui/include/**probeStatusTabUi.h** . . . ??
gui/include/**productionTabUi.h** . . . ??
gui/include/**qtHeaders.h** . . . ??
gui/include/**shutterTabUi.h** . . . ??
gui/include/**statusData.h** . . . ??
gui/include/**structureTabUi.h** . . . ??
gui/include/**tcsGui.h** . . . ??
gui/include/**thermoCubeStatusData.h** . . . ??
gui/include/**trackerPositionData.h** . . . ??
gui/include/**trackerStatusData.h** . . . ??
gui/include/**trackerStatusTabUi.h** . . . ??
gui/include/**trackerTabUi.h** . . . ??
gui/include/**trajectoryTabUi.h** . . . ??
gui/include/**videoSwitchTabUi.h** . . . ??
gui/include/**virusEnvironmentTabUi.h** . . . ??
gui/include/**weatherDataTabUi.h** . . . ??
gui/include/**webBrowserTabUi.h** . . . ??
het_lib/cJSON/include/**cJSON.h** . . . ??
het_lib/common/include/**averager.h** . . . ??
het_lib/common/include/**fits_definitions.h** . . . ??
het_lib/common/include/**fits_helper.h** . . . ??
het_lib/common/include/**fixed.h** . . . ??
het_lib/common/include/**fptime.h** . . . ??
het_lib/common/include/**het_earth_orientation.h** . . . ??
het_lib/common/include/**mainpage_doc.h** . . . ??
het_lib/common/include/**mkpath.h** . . . ??
het_lib/common/include/**modulo.h** . . . ??
het_lib/common/include/**named_reader.h** . . . ??

het_lib/common/include/**noncopyable.h** . . . ??
het_lib/common/include/**pfip_definitions.h** . . . ??
het_lib/common/include/**pool.h** . . . ??
het_lib/common/include/**quaternion.h** . . . ??
het_lib/common/include/**stacktrace.h** . . . ??
het_lib/common/include/**subsystem.h** . . . ??
het_lib/common/include/**tcs_app.h** . . . ??
het_lib/common/include/**tcs_baseclass.h** . . . ??
het_lib/common/include/**tcs_clock.h** . . . ??
het_lib/common/include/**tcs_config.h** . . . ??
het_lib/common/include/**tcs_coordinates.h** . . . ??
het_lib/common/include/**tcs_definitions.h** . . . ??
het_lib/common/include/**tcs_exception.h** . . . ??
het_lib/common/include/**tcs_fenv.h** . . . ??
het_lib/common/include/**tcs_named.h** . . . ??
het_lib/common/include/**tcs_pid.h** . . . ??
het_lib/common/include/**tcs_sla.h** . . . ??
het_lib/common/include/**tcs_socket.h** . . . ??
het_lib/common/include/**tcs_subsystem.h** . . . ??
het_lib/common/include/**tcs_telnet.h** . . . ??
het_lib/common/include/**wrap.h** . . . ??
het_lib/coord_trans/include/**amoeba.h** . . . ??
het_lib/coord_trans/include/**coords.h** . . . ??
het_lib/coord_trans/include/**pfip_coords.h** . . . ??
het_lib/coord_trans/include/**sky_fs.h** . . . ??
het_lib/coord_trans/include/**structureModel.h** . . . ??
het_lib/coord_trans/mount_model/**curveFits.h** . . . ??
het_lib/illum_lib/include/**illumination.h** . . . ??
het_lib/logger/include/**Logger.h** . . . ??
het_lib/logger/include/**loggerdb.h** . . . ??
het_lib/messaging/include/**builtin_handlers.h** . . . ??
het_lib/messaging/include/**event.h** . . . ??
het_lib/messaging/include/**message.h** . . . ??
het_lib/messaging/include/**message_handler.h** . . . ??
het_lib/messaging/include/**messaging.h** . . . ??
het_lib/messaging/include/**messaging_exception.h** . . . ??
het_lib/messaging/include/**messaging_internal.h** . . . ??
het_lib/messaging/include/**payload.h** . . . ??
het_lib/messaging/include/**receiver.h** . . . ??
het_lib/messaging/include/**sender.h** . . . ??
het_lib/messaging/include/**tap.h** . . . ??
het_lib/photometry/include/**object.h** . . . ??
het_lib/photometry/include/**photometry.h** . . . ??
het_lib/sla/include/**slalib.h** . . . ??
het_lib/sla/include/**slamac.h** . . . ??
het_lib/tthread/include/fast_mutex.h . . . 1475
het_lib/tthread/include/**threadpool.h** . . . ??
het_lib/tthread/include/**threadworker.h** . . . ??
het_lib/tthread/include/tinythread.h . . . 1476
het_lib/tthread/include/**tsdeque.h** . . . ??
het_lib/tthread/include/**tslist.h** . . . ??
het_lib/tthread/include/**tsqueue.h** . . . ??
het_lib/wfs_lib/include/**reducewfs.h** . . . ??
het_lib/wfs_lib/include/**wfs_error.h** . . . ??

het_lib/wfs_lib/include/**wfs_photometry.h** . . . ??
het_lib/wfs_lib/include/**wfslib.h** . . . ??
het_lib/wfs_lib/include/**wfslib_het.h** . . . ??
ics/**archon.h** . . . ??
ics/**config.h** . . . ??
ics/**het_saga.h** . . . ??
ics/**ics.h** . . . ??
igloo_sensors/**hwcomm.h** . . . ??
igloo_sensors/**hwdatadefs.h** . . . ??
igloo_sensors/**igloo_sensors.h** . . . ??
igloo_sensors/**igloo_sensors_config.h** . . . ??
integration/sir/Het_trajectory/**point.h** . . . ??
legacy/include/**legacy_handler_doc.h** . . . ??
legacy/include/**legacyConfig.h** . . . ??
legacy/include/**legacyDomeSubSystem.h** . . . ??
legacy/include/**legacyServer.h** . . . ??
legacy/include/**legacyShutterSubSystem.h** . . . ??
legacy/include/**legacyStructureSubSystem.h** . . . ??
legacy/include/**legacyWeatherSubSystem.h** . . . ??
logrelay/**tcs_log_relay_config.h** . . . ??
logrelay/**tcs_log_relay_driver.h** . . . ??
monitor/**tcs_monitor_config.h** . . . ??
monitor/**tcs_monitor_driver.h** . . . ??
pas/AVT_lib/include/**libavt.h** . . . ??
pas/include/**Camera.h** . . . ??
pas/include/**CameraAVT.h** . . . ??
pas/include/**CameraFLI.h** . . . ??
pas/include/**CameraPGR.h** . . . ??
pas/include/**CameraSim.h** . . . ??
pas/include/**ImageProcessor.h** . . . ??
pas/include/**LightSource.h** . . . ??
pas/include/**pas_handler_doc.h** . . . ??
pas/include/**pasConfig.h** . . . ??
pas/include/**pasDevicePipelineSubSystem.h** . . . ??
pas/include/**pasImageCleaner.h** . . . ??
pas/include/**pasImageCleanerStandard.h** . . . ??
pas/include/**pasImagePipelineSubSystem.h** . . . ??
pas/include/**pasImageProcessorPhot.h** . . . ??
pas/include/**pasImageProcessorWFS.h** . . . ??
pas/include/**pasImageStorage.h** . . . ??
pas/include/**pasImageStorageFITS.h** . . . ??
pas/include/**pasPipelineSubSystem.h** . . . ??
pas/include/**pasServer.h** . . . ??
pas/include/**pasTTLaserLightSource.h** . . . ??
pas/include/**tcs_client.h** . . . ??
pas/include/**weather_client.h** . . . ??
pfip/include/**ACQCamera.h** . . . ??
pfip/include/**ACQShutter.h** . . . ??
pfip/include/**ACQSubSystem.h** . . . ??
pfip/include/**EPICSComm.h** . . . ??
pfip/include/**FCUCommunication.h** . . . ??
pfip/include/**FCUSubSystem.h** . . . ??
pfip/include/**MAXONStatus.h** . . . ??
pfip/include/**MODBUSComm.h** . . . ??

pfip/include/**pfip_handler_doc.h** . . . ??
pfip/include/**PFIPConfig.h** . . . ??
pfip/include/**PFIPLampsDB.h** . . . ??
pfip/include/**PFIPMonitor.h** . . . ??
pfip/include/**PFIPProbeInfo.h** . . . ??
pfip/include/**pfipServer.h** . . . ??
pfip/include/**PFIPStatus.h** . . . ??
pfip/include/**Probe.h** . . . ??
pfip/include/**ProbeAssembly.h** . . . ??
pfip/MODBUS_lib/include/**config.h** . . . ??
pfip/MODBUS_lib/include/**modbus-private.h** . . . ??
pfip/MODBUS_lib/include/**modbus-rtu-private.h** . . . ??
pfip/MODBUS_lib/include/**modbus-rtu.h** . . . ??
pfip/MODBUS_lib/include/**modbus-tcp-private.h** . . . ??
pfip/MODBUS_lib/include/**modbus-tcp.h** . . . ??
pfip/MODBUS_lib/include/**modbus-version.h** . . . ??
pfip/MODBUS_lib/include/**modbus.h** . . . ??
playground/gltracker/**gltracker_config.h** . . . ??
playground/gltracker/**gltracker_driver.h** . . . ??
playground/gltracker/**minvec.h** . . . ??
playground/probepointgen/gui/**qTcsProbeAssembly.h** . . . ??
playground/sample_server/**sampleConfig.h** . . . ??
playground/sample_server/**sampleServer.h** . . . ??
plotter/**tcs_plotter_config.h** . . . ??
plotter/**tcs_plotter_driver.h** . . . ??
pointing/include/**pointing_handler_doc.h** . . . ??
pointing/include/**pointingConfig.h** . . . ??
pointing/include/**pointingServer.h** . . . ??
proxy/**tcs_proxy.h** . . . ??
proxy/**tcs_proxy_handler_doc.h** . . . ??
tcs/**dome_client.h** . . . ??
tcs/**instrument.h** . . . ??
tcs/**metrology.h** . . . ??
tcs/**pas_camera.h** . . . ??
tcs/**pfip_client.h** . . . ??
tcs/**structure_client.h** . . . ??
tcs/**tcs_driver.h** . . . ??
tcs/**tcs_handler_doc.h** . . . ??
tcs/**tcs_handlers.h** . . . ??
tcs/**tcs_server.h** . . . ??
tcs/**tcs_server_config.h** . . . ??
tcs/**tracker_client.h** . . . ??
tcs/**trajectory.h** . . . ??
tcs/**weather_client.h** . . . ??
tcs/traj_comp/**point.h** . . . ??
tcs/traj_comp/NORADlib/**norad.h** . . . ??
tcs/traj_comp/NORADlib/**norad_in.h** . . . ??
tcs/traj_comp/NORADlib/**observe.h** . . . ??
tcsconsole/**command.h** . . . ??
tcsnamed/**tcsnamed.h** . . . ??
top/**tcs_top_config.h** . . . ??
top/**tcs_top_driver.h** . . . ??
tracker/**tmcs_api.h** . . . ??
tracker/**tracker_config.h** . . . ??

tracker/**tracker_driver.h** . . . . . **??**
tracker/**tracker_handler_doc.h** . . . . . **??**
tracker/**tracker_handlers.h** . . . . . **??**
tracker/**tracker_server.h** . . . . . **??**
tracker/**tracker_socket.h** . . . . . **??**
utility_gui/include/**configureTabsDlg.h** . . . . . **??**
utility_gui/include/**droppablePlainTextEdit.h** . . . . . **??**
utility_gui/include/**guiConfig.h** . . . . . **??**
utility_gui/include/**guiPreferences.h** . . . . . **??**
utility_gui/include/**loadedUi.h** . . . . . **??**
utility_gui/include/**mainWindow.h** . . . . . **??**
utility_gui/include/**plotEventsTabUi.h** . . . . . **??**
utility_gui/include/**qtHeaders.h** . . . . . **??**
utility_gui/include/**utilityGui.h** . . . . . **??**

Chapter 7

Module Documentation

7.1 Configuration

Configuration structure for the APC Server.

7.1.1 Detailed Description

The APC server is initialized based on command line arguments and any configuration files given. This information is used to initialize the server, pipelines, light source, cameras, and image processing.

Classes

- class apcConfig

Modules

- Outlet
- Apc

7.2 Outlet

Classes

- class GroupConfig
- class CameraConfig

7.3 Apc

Classes

- class pduConfig
- class PipelineConfig

7.4 Image pdu Sub-System

Classes

- class apcPduSubSystem

7.5 Image Pdu

7.6 Payload Alignment System (APC)

Provides a control and image/data acquisition services layer for the HET Payload cameras.

7.6.1 Detailed Description

The APC Server controls all cameras on the HET Telescope Payload.

The Image Pipeline subsystem implements a series of "Image Pipe Lines", each of which can have a light source, a camera, an image processing module, and a storage module. An configuration file controls setting up these pipelines and assigns various components to each. The "work" loop of each pipeline performs the following tasks:

- Turn on and wait for the light source to warm up (optional)
- Take an image with and extract the image data from the camera
- Process the image with the desired image processing module (optional)
- Send the image to any senders (normally the HET GUI) requesting the image
- Send the processed data to any senders (normally the TCS guide correction logic) requesting the data
- Store the data in the desired format (only FITS supported at this time) (optional)
- Wait the desired configurable time between images
- Repeat the loop

Modules

- APC Message Handlers
- APC Utility Classes

7.7 APC Message Handlers

Modules

- Image Pdu

7.8 APC Utility Classes

Modules

- Configuration

  Configuration structure for the APC Server.

- Image pdu Sub-System

7.9 CAMRA Configuration Classes

Configuration class for the CAMRA application.

7.9.1 Detailed Description

The configuration is initialized based on command line arguments and any configuration files given. This information is used to initialize the application and CCD array subsystem prior to accepting any remote connections.

Classes

- class SpectrographConfig

  Spectrograph configuration class.

7.10 CAMRA Handlers

Classes

- class camra_handlers::get_ccd_temp

  Read the temperature of the given CCD and replies with the result.

- class reset
- class turn_off_leds
- class set_ccd_temp
- class set_bias_offset
- class reset_clock_voltages
- class get_clock_voltages
- class set_clock_voltages
- class get_ccd_temp
- class get_cryo_temp
- class get_cryo_pressure
- class get_heater_voltage
- class get_hardware_status
- class get_exposure_info
- class read_now
- class abort
- class expose
- class open_shutter
- class close_shutter
- class enable_shutter
- class disable_shutter
- class enable_pressure_gauges
- class disable_pressure_gauges
- class wait_for_shutter_close
- class wait_for_readout
- class wait_for_write
- class command_controller
- class reset_mux
- class reset_tcs
- class populate_onewire_device_strings
- class set_ccd_temp_xfer
- class set_cryo_temp_xfer
- class override_ifu_associations

7.11 CAMRA Hardware Classes

Classes

- class CAMRAHardware

  Hardware encapsulation class.

- class Mux

  Class corresponding to the controller multiplexing hardware.

- class Cntl

  Class corresponding the a single controller.

- class CCD

  Class corresponding to a single CCD.

- class Amp

  Class corresponding to a single CCD output amplifier.

Modules

- Array sizing constants
- Controller initialization constants
- Amplifier related constants

7.12 Array sizing constants

7.12.1 Detailed Description

These are set to the expected array size and, if greater, the corresponding hardware is flagged as offline.

The MUX_COUNT is defined here as a fallback in the event that the hardware is unavailable and is overriden by hardware discovery at runtime.

Defines

- #define **MUX_COUNT** 1
- #define **MAX_CNTL_PER_MUX** 8
- #define **MAX_CCD_PER_CNTL** 2
- #define **MAX_AMP_PER_CCD** 2
- #define **MAX_CCD_PIX_ROW** 2064
- #define **MAX_CCD_PIX_COL** 2128
- #define **OVERSCAN_COLS** 64
- #define **DEFAULT_BINNING_X** 2
- #define **DEFAULT_BINNING_Y** 1
- #define **DEFAULT_PIXELS_X** (MAX_CCD_PIX_COL/DEFAULT_BINNING_X)
- #define **DEFAULT_PIXELS_Y** (MAX_CCD_PIX_ROW/DEFAULT_BINNING_Y)
- #define **FITS_CCD_SIZE** "[1:2064,1:2064]"
- #define **FITS_CCDSEC_LOWER** "[1:2064,1:1032]"
- #define **FITS_CCDSEC_UPPER** "[1:2064,1033:2064]"
- #define **FITS_DETSEC_LOWER** FITS_CCDSEC_LOWER
- #define **FITS_DETSEC_UPPER** FITS_CCDSEC_UPPER
- #define **FITS_CONFIG0_AMPSEC** "[2064:1,1:1032]"
- #define **FITS_CONFIG1_AMPSEC** "[1:2064,1:1032]"
- #define **MAX_PIX_PER_CCD** (MAX_CCD_PIX_ROW ∗ MAX_CCD_PIX_COL)
- #define **MUX_BUFFER_STRIDE** (MAX_AMP_PER_CCD ∗ MAX_CCD_PER_CNTL ∗ MAX_CNTL_PER_MUX)
- #define **MUX_BUFFER_SIZE** (MAX_CNTL_PER_MUX ∗ MAX_CCD_PER_CNTL ∗ MAX_PIX_PER_CCD ∗ sizeof(uint16_t))
- #define **MAX_EXPOSURE_WAIT_TIME** 16
- #define **CAMRA_CRYO_TEMP_BIAS** -32.5
- #define **CAMRA_CLOCK_VOLTAGE_SENTINEL** FITS_UNSET_SENTINEL_FLOAT
- #define **CAMRA_CLOCK_VOLTAGE_VALID**(voltage) ((voltage) > CAMRA_CLOCK_VOLTAGE_SENTINEL + 1)

7.13 Controller initialization constants

7.13.1 Detailed Description

Initial sentinel value used to insert offline controllers into a map by controller ID. We assume that any controller IDs greater than this value are invalid and increment this by one each time an offline controller is encountered.

NOTE: If any of these constants move into runtime configuration then this ID should be derived at runtime. If the controller IDs from the vendor do not start and 0 and increment by 1 then we might need a larger fudge factor or a different scheme alltogether.

Defines

- #define **FUDGE** 100000
- #define **INITIAL_OFFLINE_MUX_ID** (MUX_COUNT + FUDGE)
- #define **INITIAL_OFFLINE_CNTL_ID** ((MUX_COUNT ∗ MAX_CNTL_PER_MUX) + FUDGE)
- #define **INITIAL_OFFLINE_CCD_ID** ((MUX_COUNT ∗ MAX_CNTL_PER_MUX) ∗ 2 + FUDGE)
- #define **INITIAL_OFFLINE_IFU_ID** (INITIAL_OFFLINE_CCD_ID + FUDGE)
- #define **INITIAL_OFFLINE_IFU_SLOT** (INITIAL_OFFLINE_IFU_ID + FUDGE)

7.14 Amplifier related constants

7.14.1 Detailed Description

Used for tagging output file names and filling FITS header fields.

Defines

- #define **UPPER_AMP** 0x1
- #define **UPPER_AMP_OFFSET** 1
- #define **UPPER** "U"
- #define **LOWER_AMP** 0x2
- #define **LOWER_AMP_OFFSET** 0
- #define **LOWER** "L"
- #define **LEFT** "L"
- #define **RIGHT** "R"

7.15 CCD Array Management and Recording Application (CAMRA)

7.15.1 Detailed Description

CAMRA provides a control and data acquisition services layer for multiplexed Astronomical Research Camera, Inc. CCD controller arrays. The server supports high level manipulation of the array via Handler methods accessible through the standard HET messaging and scripting interfaces.

Modules

- CAMRA Handlers
- CAMRA Classes

7.16 CAMRA Classes

Classes

- class CAMRADriver

  Encapsulates ARC CArcSys class for remote access.

- class camra_handlers::CAMRAHandler

  Base class for all remote interfaces.

Modules

- CAMRA Configuration Classes

  Configuration class for the CAMRA application.

- CAMRA Hardware Classes

7.17 TCS Fixed precision time

Time is stored and manipulated within TCS with fixed precision (see fixed.h) through the fptime data structure. It is assumed this is initialized to UTC seconds.

The fptime can be accessed through various conversion and formatting routines but any operation on time(s) must be done using fptime.

7.18 HET Earth orientation prediction

Classes

- class HETEarthOrientation

7.19 TCS Application Class

7.19.1 Detailed Description

A base class for HET processes/applications that provides basic shared functions. It provides parsing and processing for basic configuration, and logging initialization. The TCS application class also provides common signal handling for SIGTERM, SIGINT, and SIGQUIT.

Classes

- class TCSApp< Subsys, Config >

7.20 TCS Base Class

Classes

- class TCSBaseClass

7.21 Common Library

Modules

- TCS Fixed precision time
- HET Earth orientation prediction
- TCS Application Class
- TCS Base Class
- TCS Base Configuration
- TCS SubSystem Class

7.22 TCS Base Configuration

Classes

- class TCSConfig

  Configuration class for HET applcations.

7.23 Coordinate Systems and Transformations

7.23.1 Detailed Description

API for converting between the various coordinate systems and reference frames associated with the HET tracking mechanism, structure, and sensor positioning hardware.

Classes

- class altaz_coordinate
- class tracker_coordinate
- struct itf_coordinate
- class rtf_coordinate
- class tsf_coordinate

7.24 TCS SubSystem Class

Classes

- class TCSSubSystem

7.25 Coordinate Transforms and Structure Mount Model

Modules

- Coordinate Transform Library
- Structure Mount Model

7.26 Coordinate Transform Library

Classes

- class pfip_coord
- class itf_tsf

Namespaces

- namespace pt_itf
- namespace equ_hor

7.27 Structure Mount Model

Classes

- class mountModel
- class StructureModel

  Legacy Mount Model.

7.28 Common Client-Side Logging

Classes

- class Logger

7.29 Remote signalling

7.29.1 Detailed Description

Implementations of MessageHandler::process() should invoke the MESSAGE_SETJMP(Message* m) macro with the received message to set a jump marker that throws HandlerInterrupt when hit. If the macro is wrapped in a `try{}` block then the MessageHandler is able to catch the interrupt and act accordingly.

If a MessageHandler::process() method does not invoke the macro, then handling of the MESSAGE_-INTERRUPT_SIGNAL is left up to the main thread.

If the macro is invoked within a thread in scope other than immediately within the MessageHandler::process() method then undefined behavior will result.

Classes

- class InterruptHandler

Defines

- #define **HANDLER_INTERRUPT_SIGNAL** SIGUSR1
- #define **MESSAGE_SETJMP**(m)

Functions

- void **interrupt_signal_handler** (int)

7.29.2 Define Documentation

7.29.2.1 #define MESSAGE_SETJMP(m)

Value:

```
do                                                                  \
    {                                                                \
        struct sigaction sigact;                                     \
                                                                     \
        sigemptyset(&sigact.sa_mask);                                \
        sigact.sa_flags = 0;                                         \
        sigact.sa_handler = interrupt_signal_handler;                \
        sigaction(HANDLER_INTERRUPT_SIGNAL, &sigact, NULL);          \
                                                                     \
        if( m && sigsetjmp( *m->interrupt_mark(), 1 ) != 0 )         \
            MESSAGE_EXCEPTION( "Message handler %s interrupted.", m->handler().c_
      str() ); \
    }                                                                \
    while( false )
```

7.30 Messaging Library

7.30.1 Detailed Description

Authors:

Jason Ramsey, Niv Drory

The Messaging Library provides inter-process for the HET subsystems. The There are two types of communications provided:

- Bi-directional message/reply (Remote Procedure Call).

A listening server is initialized on a given endpoint with the Receiver object. One or more Sender clients may then connect to the server invoking Handler::process() methods with Message objects via the `Sender::send()` methods. Handler::process() implementations then respond to the Message.

- broadcast

The originator of a message uses an Event::broadcast() method to send the message, which is managed by the EventSource class. Event recipients register for receipt of the messages using the EventSink class.

The Messaging Library is implemented using the zeromq library (zeromq.org).

Classes

- class HelpHandler
- class StatusHandler
- class Event
- class EventSource
- class EventSourceSocket
- class EventSink
- class EventHandler
- class Message
- class MessageHandlerDescriptor
- class MessageHandler
- class Payload
- class Payload::Data
- class Receiver
- class Sender

Modules

- Remote signalling

7.31 TinyThread++ API Reference

7.31.1 Introduction

TinyThread++ is a minimal, portable implementation of basic threading classes for C++.

They closely mimic the functionality and naming of the C++11 standard, and should be easily replaceable with the corresponding std:: variants.

7.31.2 Portability

The Win32 variant uses the native Win32 API for implementing the thread classes, while for other systems, the POSIX threads API (pthread) is used.

7.31.3 Classes

In order to mimic the threading API of the C++11 standard, subsets of several classes are provided. The fundamental classes are:

- tthread::thread
- tthread::mutex
- tthread::recursive_mutex
- tthread::condition_variable
- tthread::lock_guard
- tthread::fast_mutex

7.31.4 Miscellaneous

The following special keywords are available: thread_local.

For more detailed information (including additional classes), browse the different sections of this documentation. A good place to start is: tinythread.h.

7.32 Configuration

Configuration structure for the Legacy Server.

7.32.1 Detailed Description

The Legacy server is initialized based on command line arguments and any configuration files given. This information is used to initialize the server and the legacy server(s) access.

Classes

- class legacyConfig

Modules

- Dome
- Structure
- Weather

7.33 Dome

Classes

- class DomeConfig

7.34 Dome

Classes

- class ShutterConfig

7.35 Structure

Classes

- class StructureConfig

7.36 Weather

Classes

- class WeatherConfig

7.37 Dome

Classes

- class legacyDomeSubSystem

  Provides access to the legacy Dome server.

7.38 Dome

Classes

- class OnLine

  Check to see if the legacy dome sub system is connected.

- class Abort

  Abort current Dome command (normally movement).

- class GetPosition

  Get current dome position.

- class Home

  Command the dome to find its encoder home postion.

- class MoveTo

  Move the dome to the specified azimuth.

- class SetDirection

  Set the dome movement direction.

- class SetPower

  Set the dome power state.

- class SetVelocity

  Set the dome movement velocity.

- class Reset

  Reset the dome power inverter.

- class Status

  Return the dome status information.

- class Stop

  Stop the current dome command (normally movement).

- class Test

  Start/Stop the dome movement test.

7.39 Legacy Servers Access System (Legacy)

Provides access to legacy HET control servers not re-implemented during the WFU.

7.39.1 Detailed Description

The Legacy Server provides the following Sub-Systems:

- The Dome Sub-System which accepts commands to control the Dome and the Dome Shutter.
- The Structure Sub-System which accepts commands to control the Structure movement.
- The SAMS Sub-System which accepts commands to communicate with the SAMS (Segment Alignment Maintenance System) server.
- The Weather Sub-System which accepts commands to retrieve data from the HET Weather server.

Access to these Sub-Systems is done through tcpip sockets using commands that were already present in the legacy servers.

Modules

- Class Documentation

7.40 Class Documentation

Modules

- Configuration

  Configuration structure for the Legacy Server.

- Dome
- Shutter
- Structure
- Weather

7.41 Shutter

Classes

- class legacyShutterSubSystem

  Provides access to the legacy Shutter server.

7.42 Shutter

Classes

- class OnLine

  Check to see if the legacy dome sub system is connected.

- class Abort

  Abort current Dome command (normally movement).

- class Close

  Close the shutter.

- class IssuePMACCmd

  yyy

- class Open

  Open the shutter.

- class Reset

  Reset the dome power inverter.

- class SetPower

  Set the dome power state.

- class Status

  Return the dome status information.

- class Stop

  Stop the current dome command (normally movement).

7.43 Structure

Classes

- class legacyStructureSubSystem

  Provides access to the legacy Structure server.

7.44 Structure

Classes

- class OnLine

  Check to see if the legacy dome sub system is connected.

- class MoveTo

  Move the dome to the specified azimuth.

- class MoveToStructAz

  Move the structure to the specified azimuth without mount model applied.

- class Status

  Return the dome status information.

- class Stop

  Stop the current dome command (normally movement).

- class Stow

  Stow the structure.

7.45 Weather

Classes

- class legacyWeatherSubSystem

  Provides access to the legacy HET Weather server.

7.46 Weather

Classes

- class WeatherOnLine

  Check to see if the Weather legacy server is connected.

- class GetWeatherData

  Return legacy Weather data items.

7.47 Camera

Classes

- class Camera

  Base Camera class.

Modules

- AVT Camera
- FLI Camera
- PointGrey Camera
- Simulated Camera

7.48 AVT Camera

Classes

- class CameraAVT

7.49 FLI Camera

Classes

- class CameraFLI

7.50 PointGrey Camera

Classes

- class CameraPGR

7.51 Simulated Camera

Classes

- class CameraSim

7.52 Image Processor

Classes

- class ImageProcessor

  Base Image Processor class.

Modules

- Photometry Image Processor
- Wave Front Sensor Image Processor

7.53 Light Source

Classes

- class LightSource

  Base Light Source class.

Modules

- Tip Tilt Light Source

7.54 Configuration

Configuration structure for the PAS Server.

7.54.1 Detailed Description

The PAS server is initialized based on command line arguments and any configuration files given. This information is used to initialize the server, pipelines, light source, cameras, and image processing.

Classes

- class pasConfig

Modules

- Light Source
- Outlet
- Image Processor
- Pipe Line
- Apc

7.55 Light Source

Classes

- class LightSourceConfig

7.56 Image Processor

Classes

- class ProcessorConfig

7.57 Pipe Line

Classes

- class DeviceConfig

7.58 Image Storage

Classes

- class pasImageStorage

  Base Image Storage class.

Modules

- FITS Image Storage

7.59 Image Pipeline Sub-System

Classes

- class pasDevicePipelineSubSystem
- class pasImagePipelineSubSystem

7.60 Image Pipeline

Classes

- class StartSendingImages

  Start sending images to the sender of this message.

- class StopSendingImages

  Stop sending images to the sender of this message.

- class SetPixelFormat

  Set exposure type.

- class SetReadoutSpeed

  Set readout speed.

- class SetImageType

  Set image type.

- class SetRegionOfInterest

  Set sub-image rectangle.

- class SetExposureTime

  Set exposure time.

- class SetBinning

  Set camera binning.

- class SetTargetTemperature

  Set target temperature.

- class GetTemperatures

  Get temperatures.

- class SetNumFlushes

  Set number of flushes.

- class PipelineOnLine

  Check to see if pipeline is running.

- class SetIntervalSpeed

  Set the pipeline interval speed (time between images).

- class StartPipeline

  Start the pipleine for a particular camera.

- class StopPipeline

  Stop the pipleine for a particular camera.

7.61 Photometry Image Processor

Classes

- class pasImageProcessorPhot

7.62 Wave Front Sensor Image Processor

Classes

- class pasImageProcessorWFS

7.63 FITS Image Storage

Classes

- class pasImageStorageFITS

7.64 Payload Alignment System (PAS)

Provides a control and image/data acquisition services layer for the HET Payload cameras.

7.64.1 Detailed Description

The PAS Server controls all cameras on the HET Telescope Payload.

The Image Pipeline subsystem implements a series of "Image Pipe Lines", each of which can have a light source, a camera, an image processing module, and a storage module. An configuration file controls setting up these pipelines and assigns various components to each. The "work" loop of each pipeline performs the following tasks:

- Turn on and wait for the light source to warm up (optional)
- Take an image with and extract the image data from the camera
- Process the image with the desired image processing module (optional)
- Send the image to any senders (normally the HET GUI) requesting the image
- Send the processed data to any senders (normally the TCS guide correction logic) requesting the data
- Store the data in the desired format (only FITS supported at this time) (optional)
- Wait the desired configurable time between images
- Repeat the loop

Modules

- PAS Message Handlers
- PAS Utility Classes

7.65 PAS Message Handlers

Modules

- Image Pipeline

7.66 PAS Utility Classes

Modules

- Camera
- Image Processor
- Light Source
- Configuration

  Configuration structure for the PAS Server.

- Image Storage
- Image Pipeline Sub-System

7.67 Tip Tilt Light Source

Classes

- class pasTTLaserLightSource

7.68 Acquisition and Guide Probe Assembly

7.68.1 Detailed Description

This group of classes manages the

- Acquisition camera filters and settings
- Guide Probe Assembly and movement of probes
- Main shutter for acquisition camera
- Exposures using the main shutter

The class containing the Acquisition and Guide Probe Assembly is ACQ_SubSystem, which provides the API to the operations for the pfipSserver commands. In addition to responding to change/set requests for pfipServer, the ACQ_SubSystem also maintains a background thread which reads the status flags from the hardware and updates internal status data.

Classes

- class ACQ_Camera

  A class to represent the acquisition camera.

- class ACQ_Shutter

  A class to represent the acquisition camera shutter.

- class ACQ_SubSystem

  The ACQ_SubSystem class is used to manage the PFIP hardware, including the Probe Assembly, the Acquisition Camera, and the main Shutter. It uses the EPICS protocol to communicate with the hardware.

7.69 FCUClasses

Classes that manage the components of the Facility Calibration Unit.

Classes

- class FCU_Lamp

  The FCU_Lamp class manages the state of the lamps. There are 5 lamps on each FCU slider. The FCU_Lamp class is also used for the single LDLS on the FCU.

- class FCU_Slider

  A class to represent the FCU sliders. There are two -- slider a and slider b. Each contains five positions and five lamps.

- class FCU_LDLS

  The class representing the LDLS and its state.

- class FCU_State

  A class to hold state information for the FCU. Has status of FCU Head position and lightGuide position.

- class FCU_SubSystem

  The FCU_SubSystem class is used to control the operation of the Facility Calibration Unit hardware, and provides an API to pfipServer for its management. The FCU_SubSystem contains an instance of the MODBUSComm class, which provides communication to the hardware using the MODBUS protocol. In addition to providing operational changes to the FCU, FCU_SubSystem also maintains status of all the hardware and monitors alarms and notifications about changes to the hardware and its environment. The FCU_SubSystem uses a background thread to update these status changes.

Modules

- MODBUS Utility Data Structures

7.70 MODBUS Utility Data Structures

These structures provide layout of the modbus holding registers for communication with the FCU PLC. There are MAXMODBUSREG (505) that are used. The first 150 are input parameters to the PLC used to request actions or to enable hardware. There is a large gap of unused registers (151-449). The final group of holding registers (450-504) are output information containing alarms and notifications generated by the FCU PLC.

The first group of input registers are used by the FCU Subsystem object methods. The final group are read and written by the Notification Thread of the FCU subsystem to identify state changes. There is only one (singleton) object representing the FCU _SubSystem. Since the registers are segmented in their use by the main FCU object thread and the notification thread, there is no need to provide serialized access to these items.

7.71 Prime Focus Instrument Package Server (PFIPServer)

7.71.1 Detailed Description

The PFIP server is a process that controls the PFIP hardware. PFIP is composed of several distinct hardware interfaces:

1) PFIP Acquisition and Probe System

2) Facility Calibration Unit

3) PFIP monitor

The PFIP Acquisition and Probe System is comprised of the acquisition camera, calibration wave front sensor camera, the pupil-viewer camera, filters for the cameras, and acquisition camera shutter. There are commands to change the acquisition camera filter, to open/close the acquisition camera shutter, and to expose using the acquisition shutter. The PFIP Acquisition and Probe System controls the deployment of the acquisition camera, the Calibration Wave Front Sensor (cwfs) camera, and the pupil-viewer camera. The CWFS and the Pupil-Viewer camera are mutually exclusive -- when one is deployed, the other is retracted. There are also mirrors and filters for the acquisition camera and the CWFS camera which are controlled the PFIP commands. The PFIP acquisition and probe system also has a dithering mechanism which has several different settings controlled by command.

The Guide Probe Assembly manages and coordinates movement of the guide probes and wave front sensors. There are two guide probes and two wave front sensors in the guide probe assembly. PFIP has several commands that control guide probes -- moving, homing, enabling, disabling. Each guider probe has a camera with filters that can be configured.

The PFIP Acquisition and Probe System controls the hardware using an EPICS (Experimental Physics and Industrial Control System) interface, an open source package managed by Argonne National Laboratories. The PFIP/EPICS interface is defined in the EPICSComm class.

The Facility Calibration Unit controls specialized light sources and exposures. It consists of two arrays of lamps and shutters, Slider A and Slider B. Each slider has 5 lamps and 5 shutters. There is also a single LDLS light source and and LDLS shutter. To use a lamp/slider position, the lamps or LDLS must be warmed up. Each of these light sources have a configured warmup time that is conveyed to the PFIP using a command. The light sources on/off is also controlled by commands. To make an exposure using one of the lamps on a slider, the FCU arm should be deployed (unless overridden), the slider should be moved into the correct position, and an exposure time specified. There is also a another light control, the lightGuide, which can be set to Red or Blue. The FCU commands in PFIP are used to control the slider position, the lamps, and the shutters.

The FCU also responds to the PFIP with information about the safety and status of the hardware, providing alarms and warnings of unusual conditions which should be reported. The PFIP control system communicates with the FCU hardware PLC using a Modbus interface, which is defined in the MODBUSComm class. The modbus library is provided by libmodbus.org, which is an open-source modbus protocol imnplementation.

Finally, PFIP communicates with another piece of hardware that provides monitoring of the state of the PFIP acquisition and probe system, issuing alarms and warnings about the environment and health of the hardware. This is also communicated using Modbus.

Modules

- PFIP Configuration Settings

  The PFIP configuration file contains the PFIP configuration variables. There is a standard set of configuration variables defined by the TCSConfig object. The PFIPConfig object extends TCSConfig with the

following additional variables:.

- PFIPServer Classes

  This group defines the classes that implement PFIPServer.

- PFIP Interface Operations and Events

  This section lists the operations provided by the PFIP server. They are divided into two types -- those controlling primarily the acquisition camera/guide probe controls (ACQSystem Messages) and the Facility Calibration Unit (FCU Messages).

7.72 PFIP Configuration Settings

The PFIP configuration file contains the PFIP configuration variables. There is a standard set of configuration variables defined by the TCSConfig object. The PFIPConfig object extends TCSConfig with the following additional variables:.

7.73 PFIPServer Classes

This group defines the classes that implement PFIPServer.

Classes

- class EPICSComm
- class PFIPConfig

  A class to handle the PFIP Configuration file and settings, extending the base TCSConfig class.

- class Alarm_Monitor
- class PFIP_Monitor
- class pfipServer
- class PFIPHandler

  A class derived from Handler to encompass all the message handlers for PFIP server.

- class VENCHandler

  A class derived from PFIPHandler to provide message handlers for FCU commands.

Modules

- Acquisition and Guide Probe Assembly
- FCUClasses

  Classes that manage the components of the Facility Calibration Unit.

- Probe Assembly classes
- EPICSVariables

  All the EPICS variables for PFIP Acquisition and Probe System.

7.74 PFIP Interface Operations and Events

This section lists the operations provided by the PFIP server. They are divided into two types -- those controlling primarily the acquisition camera/guide probe controls (ACQSystem Messages) and the Facility Calibration Unit (FCU Messages).

7.75 Acquisition and Probe Assembly

These are messages to control the PFIP Acquisition System. Unless otherwise noted, the operations commanded by the messages are synchronous and blocking. They do not return until the operation has completed, and only one of the commands can be in effect at a time.

7.76 Facility Calibration Unit

Handlers for the Facility Calibration Unit sub-system commands. These methods are all derived from the PFIPHandler method, which is derived from Handler.

7.77 Probe Assembly classes

7.77.1 Detailed Description

The probe assembly consists of a circular track on which there are 4 probes. Each probe has approximately 180 degrees of movement, distributed evenly around the circular track, such that each probe's path overlaps each adjacent probe's path. The four probes consist of two guider probes (GUIDER1 and GUIDER2) and two wave front sensor probes (WFS1 and WFS2). The probe identified as GUIDER1 is arbitrarily defined to start in the first quadrant of the circle, followed by WFS1 in quadrant 2, GUIDER2 in quadrant 3, and WFS2 in quadrant 4. The probes are assigned ranges based on configuration information in het_lib/coord_trans/include/pfip_coords.h, but they roughtly correspond to the following degree arcs: GUIDER2: 46-223 GUIDER1: 226-403 WFS2: 136-313 WFS1: 316-493

Each probe can move clockwise or counter-clockwise in its range. So, each probe's path can overlap that probe path on either side. The probe move algorithm examines the source location of each probe and determines if either of the adjacent probes must move first in order to allow the other probe to reach its location. Then, it builds a dependency list of the order in which the probes must move in order to avoid collisions. There is a possibility that the dependency list is circular. In this case, one of the probes is selected to move partway to its destination, and then the dependency list is re-calculated to see if the circular dependency has been removed. The algorithm attempts the dependency list calculation 4 times before finally giving up. As part of the dependency calculation, the algorithm also allows multiple probes to move at the same time, if they will not get in each other's way.

Each probe has an arm with a camera sensor at each end. The point to which the arm extends is the target of the probe. Each arm has about 20 degrees of arc.

Classes

- class PFIPProbeInfo

  A class encapsulating hardware status for PFIP probes.

- class Probe
- class ProbeAssembly

  A class to represent the probe assembly. Handles the coordinated movement and collision avoidance for the four probes.

Enumerations

- enum WhichMotor { **PROBE** = 0, **ARM** }

7.77.2 Enumeration Type Documentation

7.77.2.1 enum WhichMotor

A class to represent a probe position

7.78 EPICSVariables

All the EPICS variables for PFIP Acquisition and Probe System.

7.79 Configuration

Configuration structure for the Sample Server.

7.79.1 Detailed Description

The Sample server is initialized based on command line arguments and any configuration files given. This information is used to initialize the server and the sample server(s) access.

Classes

- class sampleConfig

7.80 Sample Servers Access System (Sample)

Provides access to sample HET control servers not re-implemented during the WFU.

7.80.1 Detailed Description

The Sample Server provides the following Sub-Systems:

- The Dome Sub-System which accepts commands to control the Dome and the Dome Shutter.
- The Structure Sub-System which accepts commands to control the Structure movement.
- The SAMS Sub-System which accepts commands to communicate with the SAMS (Segment Alignment Maintenance System) server.
- The Weather Sub-System which accepts commands to retrieve data from the HET Weather server.

Access to these Sub-Systems is done through tcpip sockets using commands that were already present in the sample servers.

Modules

- Message Handler Documentation
- Class Documentation

7.81 Message Handler Documentation

7.82 Class Documentation

Modules

- Configuration

  Configuration structure for the Sample Server.

7.83 Configuration

Configuration structure for the Pointing Server.

7.83.1 Detailed Description

The Pointing server is initialized based on command line arguments and any configuration files given. This information is used to initialize the server and the pointing server(s) access.

Classes

- class pointingConfig

7.84 Pointing Servers Access System (Pointing)

Provides access to pointing HET control servers not re-implemented during the WFU.

7.84.1 Detailed Description

The Pointing Server provides the following Sub-Systems:

- The Dome Sub-System which accepts commands to control the Dome and the Dome Shutter.
- The Structure Sub-System which accepts commands to control the Structure movement.
- The SAMS Sub-System which accepts commands to communicate with the SAMS (Segment Alignment Maintenance System) server.
- The Weather Sub-System which accepts commands to retrieve data from the HET Weather server.

Access to these Sub-Systems is done through tcpip sockets using commands that were already present in the pointing servers.

Modules

- Class Documentation

7.85 Class Documentation

Modules

- Configuration

  Configuration structure for the Pointing Server.

7.86 Telescope Control System (TCS)

TCS provides a control and data acquisition services layer for the TCS hardware sub-systems.

7.87 Tracker Motion Control System (TMCS) API

7.87.1 Detailed Description

7.87.2 SET_TRAJ

Usage:

```
SET_TRAJ cmd_idx tset_num time X Y Z rho theta phi
```

Sends a single, serial-indexed, time-based trajectory coordinate to the Tracker CS. This command will typically be sent in batches, as part of a set of coordinates which represent a full trajectory.

The coordinates (X, Y, Z, rho, theta, phi) are always specified as real tracker frame coordinates (RTF coordinate frame), which implies that the TCS has applied a mount model to take care of things like structure deformation, and the Tracker CS will apply a mount model to take care of things like bridge deformation, encoder zero-points and scale factors.

The index (cmd_idx) parameter has nothing to do with the trajectory coordinate, or even the position of a particular coordinate within a trajectory set. The tracker controller uses this index as part of its acknowledgement reply back to the TCS, giving the TCS confirmation that the tracker controller successfully received a particular coordinate. The tracker controller should instead use the time parameter to organize the coordinate within its system memory. If a SET_TRAJ command is received by the tracker controller which contains a time that it had already received in the past, the latest command data will always supersede any previous data.

The trajectory set number parameter (tset_num) indicates what trajectory set this point belongs to (tset_num = 1 to 999), or if the point is a 'repeated' point (complimented by tset_num = -999 to -1). A value of 1 indicates the current running trajectory set, or, if no trajectory is currently running, the next trajectory to run. A number higher than this (ie, 2 to 999) indicates trajectories that would follow the current trajectory. (For practical purposes, there are no short-term plans to have more than 1 trajectory set loaded in the TC at any given time. However, to provide flexibility for certain engineering modes in the future, the API will support up to 999 separate sequential trajectories.) A repeated point (indicated by a negative tset_num) is one that was sent previously from the TCS, but the TCS didn't receive acknowledgement ("SET_TRAJ 01223 OK") back from the TC, therefore it is resending the point. The TC should insert the repeated point in its collection of trajectory points and not purge the points that follow it (as it would when receiving a non-repeated point).

The time parameter is index time (number of seconds from an established 'zero point'). Index time will include any leap seconds that have occurred since the last crossing of the established zero point.

7.87.2.1 Reference

- cmd_idx = max 5 ascii bytes, followed by space (ASCII 32)
- tset_num = max 4 ascii bytes, followed by space (ASCII 32)
- time = max 10 ascii bytes, followed by space (ASCII 32)
- X = max 9 ascii bytes, followed by space (ASCII 32)
- Y = max 9 ascii bytes, followed by space (ASCII 32)
- Z = max 9 ascii bytes, followed by space (ASCII 32)
- Rho = max 7 ascii bytes, followed by space (ASCII 32)

- Theta = max 7 ascii bytes, followed by space (ASCII 32)
- Phi = max 7 ascii bytes, followed by space (ASCII 32)

7.87.2.2 Example

```
SET_TRAJ 01223 1 86399.0123 +234.654 -1200.987 123.455 -15.670 +2.0099 -4.5324 [
    Padding]<CR><LF>
```

7.87.2.3 Returns

Success:

```
SET_TRAJ 01223 OK [Padding]<CR><LF>
```

Error(s):

```
SET_TRAJ 01223 ERROR [Padding]<CR><LF>
```

7.87.2.4 Notes

- time will translate in to a double, and indicates the number of seconds from an established zero point
- tset_num will translate into a signed long, no units (sequence), with a range of -999 to 999 (however, 0 is invalid)
- cmd_idx will translate into a long, no units (sequence), with a range of 0 to 99999
- X will translate into a double, units are millimeters, with a range of +/- 1960.000 mm
- Y will translate into a double, units are millimeters, with a range of +/- 1940.000 mm
- Z will translate into a double, units are millimeters, with a range of -75.000 to +290.000
- Rho will translate into a double, units are degrees, with a range of +/- 27.000 deg
- Theta will translate into a double, units are degrees, with a range of +/- 9.5000 deg
- Phi will translate into a double, units are degrees, with a range of +/- 9.5000 deg

7.87.3 ABORT

Usage:

```
ABORT cmd_idx tset_num
```

The tracker controller will abort an INIT, MOVA_RTF, MOVR_RTF, MOVA_TSF, MOVR_TSF or a Trajectory move and gracefully bring the tracker to a halt (if it is moving), and put it into its Idle state. In addition, if in trajectory mode, the tracker controller will erase from its memory any previously received trajectory information corresponding to the tset_num. The tset_num is an optional parameter that applies only to Trajectory and was added to allow multiple trajectories to be loaded into TMCS (tracker controller). During a Trajectory, the currently loaded Trajectory will be deleted if tset_num is greater than or equal to 0 and less than 1000.

Multi-trajectories are not supported at this time. Non-trajectory moves are not required to send the tset_num parameter.

The index (cmd_idx) parameter has nothing to do with the command. The tracker controller uses this index as part of its acknowledgement reply back to the TCS, giving the TCS confirmation that the tracker controller successfully received the command.

7.87.3.1 Reference

- cmd_idx = max 5 ascii bytes, followed by space (ASCII 32)
- tset_num = max 4 ascii bytes, followed by space (ASCII 32)

7.87.3.2 Example

```
ABORT 01234 44 [Padding]<CR><LF>
```

7.87.3.3 Returns

Success:

```
ABORT 01234 OK [Padding]<CR><LF>
```

Error(s):

```
ABORT 01234 ERROR [Padding]<CR><LF>
```

7.87.3.4 Notes

- cmd_idx will translate into a long, no units (sequence), with a range of 0 - 99999
- tset_num will translate into a signed long, no units (sequence), with a range of -999 to 999

7.87.4 MOVR_RTF

Usage:

```
MOVR_RTF cmd_idx X Y Z rho theta phi vel
```

Initiates a relative (incremental) move from the current location in real tracker frame coordinates (RTF coordinate frame), at the requested velocity.

If the tracker controller can initiate the move (move parameters are correct and the controller is in the proper state, etc), then the tracker controller will respond with the "Command Success" response, and initiates the move. If the tracker controller is not ready to initiate the move, it will respond with the "Command Error" response, which includes a cmd_error_num (see appendix for definition). If the move completes successfully, the tracker controller will send the "Move Success" response. If not, it will send the "Move Error" response, which includes a move_error_num (see appendix for definition).

The vel parameter will be one of the following: SLEW_SLOW, SLEW_FAST, TRK_SLOW, TRK_FAST. Each of these corresponds to a set of parameters, each set containing velocities for each axis. See appendix on parameters for more detail.

The index (cmd_idx) parameter has nothing to do with the command. The tracker controller uses this index as part of its acknowledgement reply back to the TCS, giving the TCS confirmation that the tracker controller successfully received the command.

7.87.4.1 Reference

- cmd_idx = max 5 ascii bytes, followed by space (ASCII 32)
- X = max 9 ascii bytes, followed by space (ASCII 32)
- Y = max 9 ascii bytes, followed by space (ASCII 32)
- Z = max 8 ascii bytes, followed by space (ASCII 32)
- rho = max 7 ascii bytes, followed by space (ASCII 32)
- theta = max 8 ascii bytes, followed by space (ASCII 32)
- phi = max 8 ascii bytes, followed by space (ASCII 32)
- vel = max 9 ascii bytes, followed by space (ASCII 32)

7.87.4.2 Example

```
MOVR_RTF 01234 +12.434 333.123 123.456 -4.564 +5.1377 -4.4237 SLEW_SLOW [Padding
    ]<CR><LF>
```

7.87.4.3 Returns

Command Success:

```
MOVR_RTF 01234 OK [Padding]<CR><LF>
```

Move Success:

```
MOVR_RTF 01234 COMPLETE [Padding]<CR><LF>
```

Command Error(s):

```
MOVR_RTF 01234 C_ERROR cmd_error_num [Padding]<CR><LF>
```

Move Error(s):

```
MOVR_RTF 01234 M_ERROR move_error_num [Padding]<CR><LF>
```

7.87.4.4 Notes

- cmd_idx will translate into a long, no units (sequence), with a range of 0 - 99999
- X will translate into a double, units are millimeters, with a range of +/- 3920.000 mm
- Y will translate into a double, units are millimeters, with a range of +/- 3880.000 mm
- Z will translate into a double, units are millimeters, with a range of +/- 365.000 mm
- rho will translate into a double, units are degrees, with a range of +/- 54.000 deg
- theta will translate into a double, units are degrees, with a range of +/- 19.0000 deg
- phi will translate into a double, units are degrees, with a range of +/- 19.0000 deg
- vel will translate into either the string SLEW_SLOW, SLEW_FAST, TRK_SLOW, or TRK_FAST
- cmd_error_num = max 5 ascii bytes, followed by space (ASCII 32)
- move_error_num = max 5 ascii bytes, followed by space (ASCII 32)

7.87.5 MOVA_RTF

Usage:

```
MOVA_RTF cmd_idx X Y Z rho theta phi vel
```

Initiates an absolute move from the current location in real tracker frame coordinates (RTF coordinate frame) at the velocity specified.

If the tracker controller can initiate the move (move parameters are correct and the controller is in the proper state, etc), then the tracker controller will respond with the "Command Success" response, and initiates the move. If the tracker controller is not ready to initiate the move, it will respond with the "Command Error" response, which includes a cmd_error_num (see appendix for definition). If the move completes successfully, the tracker controller will send the "Move Success" response. If not, it will send the "Move Error" response, which includes a move_error_num (see appendix for definition).

The vel parameter will be one of the following: SLEW_SLOW, SLEW_FAST, TRK_SLOW, TRK_FAST. Each of these corresponds to a set of parameters, each set containing velocities for each axis. See appendix on parameters for more detail.

The index (cmd_idx) parameter has nothing to do with the command. The tracker controller uses this index as part of its acknowledgement reply back to the TCS, giving the TCS confirmation that the tracker controller successfully received the command.

7.87.5.1 Reference

- cmd_idx = max 5 ascii bytes, followed by space (ASCII 32)
- X = max 9 ascii bytes, followed by space (ASCII 32)
- Y = max 9 ascii bytes, followed by space (ASCII 32)
- Z = max 8 ascii bytes, followed by space (ASCII 32)
- rho = max 7 ascii bytes, followed by space (ASCII 32)
- theta = max 7 ascii bytes, followed by space (ASCII 32)
- phi = max 7 ascii bytes, followed by space (ASCII 32)
- vel = max 9 ascii bytes, followed by space (ASCII 32)

7.87.5.2 Example

```
MOVA_RTF 01234 +12.434 333.123 123.456 -4.5604 +5.1003 -4.4023 TRK_FAST [Padding
    ]<CR><LF>
```

7.87.5.3 Returns

Command Success:

```
MOVA_RTF 01234 OK [Padding]<CR><LF>
```

Move Success:

```
MOVA_RTF 01234 COMPLETE [Padding]<CR><LF>
```

Command Error(s):

```
MOVA_RTF 01234 C_ERROR cmd_error_num [Padding]<CR><LF>
```

Move Error(s):

```
MOVA_RTF 01234 M_ERROR move_error_num [Padding]<CR><LF>
```

7.87.5.4 Notes

- cmd_idx will translate into a long, no units (sequence), with a range of 0 - 99999
- X will translate into a double, units are millimeters, with a range of +/- 1960.000 mm
- Y will translate into a double, units are millimeters, with a range of +/- 1940.000 mm
- Z will translate into a double, units are millimeters, with a range of -91.000 to +363.000 mm
- rho will translate into a double, units are degrees, with a range of +/- 27.000 deg
- theta will translate into a double, units are degrees, with a range of +/- 9.5000 deg
- phi will translate into a double, units are degrees, with a range of +/- 9.5000 deg
- vel will translate into either the string SLEW_SLOW, SLEW_FAST, TRK_SLOW, or TRK_FAST
- cmd_error_num = max 5 ascii bytes, followed by space (ASCII 32)
- move_error_num = max 5 ascii bytes, followed by space (ASCII 32)

7.87.6 MOVR_TSF

Usage:

```
MOVR_TSF cmd_idx X Y \rho H1 H2 H3 H4 H5 H6 vel
```

Initiates a relative (incremental) move from the current location in tracker coordinates at the velocity specified.

If the tracker controller can initiate the move (move parameters are correct and the controller is in the proper state, etc), then the tracker controller will respond with the "Command Success" response, and initiates the move. If the tracker controller is not ready to initiate the move, it will respond with the "Command Error" response, which includes a cmd_error_num (see appendix for definition). If the move completes successfully, the tracker controller will send the "Move Success" response. If not, it will send the "Move Error" response, which includes a move_error_num (see appendix for definition).

The vel parameter will be one of the following: SLEW_SLOW, SLEW_FAST, TRK_SLOW, TRK_FAST. Each of these corresponds to a set of parameters, each set containing velocities for each axis. See appendix on parameters for more detail.

The index (cmd_idx) parameter has nothing to do with the command. The tracker controller uses this index as part of its acknowledgement reply back to the TCS, giving the TCS confirmation that the tracker controller successfully received the command.

7.87.6.1 Reference

- cmd_idx = max 5 ascii bytes, followed by space (ASCII 32)
- X = max 9 ascii bytes, followed by space (ASCII 32)
- Y = max 9 ascii bytes, followed by space (ASCII 32)
- rho = max 7 ascii bytes, followed by space (ASCII 32)
- H1 = max 8 ascii bytes, followed by space (ASCII 32)
- H2 = max 8 ascii bytes, followed by space (ASCII 32)
- H3 = max 8 ascii bytes, followed by space (ASCII 32)
- H4 = max 8 ascii bytes, followed by space (ASCII 32)
- H5 = max 8 ascii bytes, followed by space (ASCII 32)
- H6 = max 8 ascii bytes, followed by space (ASCII 32)
- vel = max 9 ascii bytes, followed by space (ASCII 32)

7.87.6.2 Example

```
MOVR_TSF 01234 +1.434 3.123 2.456 -4.5064 +5.1066 -4.4023 2.1306 -4.2033 2.2882
     SLEW_FAST [Padding]<CR><LF>
```

7.87.6.3 Returns

Command Success:

```
MOVR_TSF 01234 OK [Padding]<CR><LF>
```

Move Success:

```
MOVR_TSF 01234 COMPLETE [Padding]<CR><LF>
```

Command Error(s):

```
MOVR_TSF 01234 C_ERROR cmd_error_num [Padding]<CR><LF>
```

Move Error(s):

```
MOVR_TSF 01234 M_ERROR move_error_num [Padding]<CR><LF>
```

7.87.6.4 Reference

- cmd_idx will translate into a long, no units (sequence), with a range of 0 - 99999
- X will translate into a double, units are millimeters, with a range of +/- 3920.000 mm
- Y will translate into a double, units are millimeters, with a range of +/- 3880.000 mm
- rho will translate into a double, units are degrees, with a range of +/- 54.000 deg
- H1 will translate into a double, units are millimeters, with a range of +/- 365.0000 mm

- H2 will translate into a double, units are millimeters, with a range of +/- 365.0000 mm
- H3 will translate into a double, units are millimeters, with a range of +/- 365.0000 mm
- H4 will translate into a double, units are millimeters, with a range of +/- 365.0000 mm
- H5 will translate into a double, units are millimeters, with a range of +/- 365.0000 mm
- H6 will translate into a double, units are millimeters, with a range of +/- 365.0000 mm
- vel will translate into either the string SLEW_SLOW, SLEW_FAST, TRK_SLOW, or TRK_FAST
- cmd_error_num = max 5 ascii bytes, followed by space (ASCII 32)
- move_error_num = max 5 ascii bytes, followed by space (ASCII 32)

7.87.7 MOVA_TSF

Usage:

```
MOVA_TSF cmd_idx X Y rho H1 H2 H3 H4 H5 H6 vel
```

Initiates an absolute move in tracker coordinates at either slew speed, track speed, or else at the velocity specified.

If the tracker controller can initiate the move (move parameters are correct and the controller is in the proper state, etc), then the tracker controller will respond with the "Command Success" response, and initiates the move. If the tracker controller is not ready to initiate the move, it will respond with the "Command Error" response, which includes a cmd_error_num (see appendix for definition). If the move completes successfully, the tracker controller will send the "Move Success" response. If not, it will send the "Move Error" response, which includes a move_error_num (see appendix for definition).

The vel parameter will be one of the following: SLEW_SLOW, SLEW_FAST, TRK_SLOW, TRK_FAST. Each of these corresponds to a set of parameters, each set containing velocities for each axis. See appendix on parameters for more detail.

The index (cmd_idx) parameter has nothing to do with the command. The tracker controller uses this index as part of its acknowledgement reply back to the TCS, giving the TCS confirmation that the tracker controller successfully received the command.

7.87.7.1 Reference

- cmd_idx = max 5 ascii bytes, followed by space (ASCII 32)
- X = max 11 ascii bytes, followed by space (ASCII 32)
- Y = max 11 ascii bytes, followed by space (ASCII 32)
- rho = max 11 ascii bytes, followed by space (ASCII 32)
- H1 = max 8 ascii bytes, followed by space (ASCII 32)
- H2 = max 8 ascii bytes, followed by space (ASCII 32)
- H3 = max 8 ascii bytes, followed by space (ASCII 32)
- H4 = max 8 ascii bytes, followed by space (ASCII 32)
- H5 = max 8 ascii bytes, followed by space (ASCII 32)
- H6 = max 8 ascii bytes, followed by space (ASCII 32)
- vel = max 9 ascii bytes, followed by space (ASCII 32)

7.87.7.2 Example

```
MOVA_TSF 01234 +1.434 3.123 2.456 -4.5064 +5.1003 -4.4023 2.1003 -4.2033 2.2002
     TRK_SLOW [Padding]<CR><LF>
```

7.87.7.3 Returns

Command Success:

```
MOVA_TSF 01234 OK [Padding]<CR><LF>
```

Move Success:

```
MOVA_TSF 01234 COMPLETE [Padding]<CR><LF>
```

Command Error(s):

```
MOVA_TSf 01234 C_ERROR cmd_error_num [Padding]<CR><LF>
```

Move Error(s):

```
MOVA_TSF 01234 M_ERROR move_error_num [Padding]<CR><LF>
```

7.87.7.4 Notes

- cmd_idx will translate into a long, no units (sequence), with a range of 0 - 99999
- X will translate into a double, units are millimeters, with a range of +/- 1960.000 mm
- Y will translate into a double, units are millimeters, with a range of +/- 1940.000 mm
- rho will translate into a double, units are degrees, with a range of +/- 27.000 deg
- H1 will translate into a double, units are millimeters, with a range of -65.0000 to +280.0000 mm
- H2 will translate into a double, units are millimeters, with a range of -65.0000 to +280.0000 mm
- H3 will translate into a double, units are millimeters, with a range of -65.0000 to +280.0000 mm
- H4 will translate into a double, units are millimeters, with a range of -65.0000 to +280.0000 mm
- H5 will translate into a double, units are millimeters, with a range of -65.0000 to +280.0000 mm
- H6 will translate into a double, units are millimeters, with a range of -65.0000 to +280.0000 mm
- vel will translate into either the string SLEW_SLOW, SLEW_FAST, TRK_SLOW, or TRK_FAST
- cmd_error_num = max 5 ascii bytes, followed by space (ASCII 32)
- move_error_num = max 5 ascii bytes, followed by space (ASCII 32)

7.87.8 INIT

Usage:

```
INIT cmd_idx
```

Instructs the tracker controller to calibrate the incremental sensors on the Upper X, Lower X, Y and rho axis.

If the tracker controller can initiate the INIT comand (parameters are correct and the controller is in the proper state, etc), then the tracker controller will respond with the "Command Success" response, and initiates the command. If the the tracker controller is not ready to initiate the move, it will respond with the "Command Error" response, which includes a cmd_error_num (see appendix for definition). When the command completes, if completed successfully, the tracker controller will send the "Move Success" response. If not, it will send the "Move Error" response, which includes a move_error_num (see appendix for definition).

The index (cmd_idx) parameter has nothing to do with the command. The tracker controller uses this index as part of its acknowledgement reply back to the TCS, giving the TCS confirmation that the tracker controller successfully received the command.

Every second during calibration, a 'progess' message will be sent to TCS and to the Logger. These transmission uses the command type 'MSG'. See section tmcs_msg MSG for example: Format is "MSG %d %d %9.3f INIT_PROGRESS: %10.6f %5.1f %1d %1d %1d %1d [Padding]<CR><LF>", id,level,eventTime,time_left,rho_index_offset,UX,LX,Y,Rho

- id is 5 ascii chars set to constant 3000, not really needed since no ACK is used.
- level is priority level, set by TMCS via Control Desk 0 - 100
- eventTime is index time
- time_left is time to left to finish calibration in seconds
- rho_index_offset is calibration value used to seed algoritm
- UX is upper X axis completion flag, 1 = passed, 0 = failed or imcomplete
- LX is lower X axis completion flag, 1 = passed, 0 = failed or imcomplete
- Y is Y axis completion flag, 1 = passed, 0 = failed or imcomplete
- Rho is Rho axis completion flag, 1 = passed, 0 = failed or imcomplete

7.87.8.1 Example

```
INIT 01234 [Padding]<CR><LF>
```

7.87.8.2 Returns

Command Success:

```
INIT 01234 OK [Padding]<CR><LF>
```

Init Success:

```
INIT 01234 COMPLETE [Padding]<CR><LF>
```

Command Error(s):

```
INIT 01234 C_ERROR cmd_error_num [Padding]<CR><LF>
```

Move Error(s):

```
INIT 01234 M_ERROR move_error_num [Padding]<CR><LF>
```

7.87.8.3 Notes

- cmd_idx will translate into a long, no units (sequence), with a range of 0 - 99999
- cmd_error_num = max 5 ascii bytes, followed by space (ASCII 32)
- move_error_num = max 5 ascii bytes, followed by space (ASCII 32)

7.87.9 Watchdog Command

Usage:

```
ACK cmd_idx
```

The TCS and tracker controller will each transmit a simple "ACK" command at a frequency of 1 Hz. This signal provides the other systems with the knowledge that the system is up and running. If the tracker controller is currently in a moving state, and it stops receiving the watchdog signal from the TCS, it will immediately halt motion (gracefully) and hold position.

7.87.9.1 Example

```
ACK cmd_idx [Padding]<CR><LF>
```

7.87.9.2 Returns

```
ACK cmd_idx state X Y Z R T P SLX_b SUX_b SY_b TLX_b TUX_b TY_b    [Padding]<CR>
     <LF>
```

7.87.9.3 Notes

- cmd_idx will translate into a long, no units (sequence), with a range of 0 - 99999
- state indicates the current state of the Tracker CS
- tracker_state_defs.init_idle = 110; // Waiting for INIT command, NOT CALIBRATED
- tracker_state_defs.init_track_drive_moving = 152; // Moving during INIT command
- tracker_state_defs.init_track_drive_stop = 165; // Request to stop has been issued, process ongoing, NOT stopped yet.
- tracker_state_defs.tracker_ready = 210; // Calibrated and ready to receive a command (Eng. move or trajectory) and Stopped
- tracker_state_defs.slew_drive_moving = 222; // Slew drive moving

- tracker_state_defs.slew_drive_holding = 223; // End of slew move, holding at current position
- tracker_state_defs.slew_drive_stop = 235; // Request to stop has been issued, process ongoing, NOT stopped yet.
- tracker_state_defs.track_drive_moving = 252; // Track drive in Engineering move beginning of trajectory
- tracker_state_defs.track_drive_tracking = 254; // Track drive operating during a Trajectory
- tracker_state_defs.track_drive_holding = 253; // End of track move, holding at current position
- tracker_state_defs.track_drive_stop = 265; // Request to stop has been issued, process ongoing, NOT stopped yet.
- tracker_state_defs.e_stop = 999; % for now // Emergency stop enabled
- tracker_state_defs.fault_state_offset = 400; // Added to commands above when an error occurs.
- X (current X pos) will translate into a double, units are millimeters, with a range of +/- 1960.000 mm (in RTF)
- Y (current Y pos) will translate into a double, units are millimeters, with a range of +/- 1940.000 mm (in RTF)
- Z (current Z pos) will translate into a double, units are millimeters, with a range of -91.000 to +363.000 mm (in RTF)
- R (current rho pos) will translate into a double, units are degrees, with a range of +/- 27.000 deg (in RTF)
- T (current theta pos) will translate into a double, units are degrees, with a range of +/- 9.5000 deg (in RTF)
- P (current phi pos) will translate into a double, units are degrees, with a range of +/- 9.5000 deg (in RTF)
- SLX_b Slew LX brake state (0=engaged, 1=released)
- SUX_b Slew UX brake state (0=engaged, 1=released)
- SY_b Slew Y brake state (0=engaged, 1=released)
- TLX_b Track LX brake state (0=engaged, 1=released)
- TUX_b Track UX brake state (0=engaged, 1=released)
- TY_b Track Y brake state (0=engaged, 1=released)

7.87.10 MSG

Usage:

```
MSG msg_idx msg_type event_indexTime MessageString
```

The Tracker can send an event message to the TCS at any time. This mechanism will be used for general events, warnings, alarms, errors, etc. When and event occurs, the tracker will send one notification message with the index time that the event occured.

The msg_idx is a sequential number generated by the tracker to identify the message. If the message is an error and must be subsequently cleared by the TCS, then this msg_idx will be used with the complimentary CLR command (see below).

The msg_type parameter indicates what type of event this message is for, according to this table:

msg_type | Meaning

- 0 General event - Message will be displayed on GUI, but no action will occur
- 1 Warning - Warning will be displayed on GUI, but no action will occur
- 2 Error - Error will be displayed on GUI. Error must be cleared before tracker can be used (see CLR cmd below)

7.87.10.1 Reference

- msg_idx = max 5 ascii bytes, followed by space (ASCII 32)
- msg_type = max 1 ascii bytes, followed by space (ASCII 32)
- event_indexTime = max 9 ascii bytes, followed by space (ASCII 32)
- message = max 114 ascii bytes, followed by space (ASCII 32)

7.87.10.2 Format

```
MSG msg_idx msg_type event_indexTime Message from Tracker [Padding]<CR><LF>
```

Event

```
MSG 12345 0 14323.4321 Track complete, holding position [Padding]<CR><LF>
```

Warning

```
MSG 12346 1 14323.4321 Slew Y motor temp approaching threshold [Padding]<CR><LF>
```

Error

```
MSG 12347 2 14323.4321 Slew Y motor temp exceeds threshold [Padding]<CR><LF>
```

Severe Error

```
MSG 12348 3 14323.4321 Slew Y motor on fire...failure [Padding]<CR><LF>
```

Init Progress

```
MSG 3000 0 10523.000 INIT_PROGRESS:   10.0   -0.1 0 0 0 0 [Padding]<CR><LF>
```

7.87.10.3 Returns

The TCS will acknowledge the receipt of all messages with the following:

```
MSG msg_idx OK [Padding]<CR><LF>
```

7.87.10.4 tmcs_Notes

- msg_idx will translate into a long, no units (sequence), with a range of 0 - 99999
- msg_type will translate into a long, no units (sequence), with a range of 0 - 9
- event_indexTime will translate into a double, with a range of 0.000 to 86400.000

7.87.11 CLR

Usage:

```
CLR cmd_idx msg_idx
```

The clear (CLR) command is used to clear a previously sent error message according to msg_idx. This command should only be sent once the error condition has been remedied (often involving human intervention). If the error condition has not been remedied, then the tracker should reply with "CLR msg_idx ERROR" . If the tracker assesses that the error condition has been successfully remedied, then the tracker should reply with "CLR msg_idx OK" indicating that the message was successfully cleared and it is no longer in the error state.

7.87.11.1 Reference

- msg_idx = max 5 ascii bytes, followed by space (ASCII 32)

7.87.11.2 Format

```
CLR cmd_idx msg_idx [Padding]
```

7.87.11.3 Example

```
CLR 01234 12347 [Padding]<CR><LF>
```

7.87.11.4 Returns

If the clearing of the error was successful and the tracker is no longer in the error state:

```
CLR cmd_idx OK [Padding]<CR><LF>
CLR 01234 OK [Padding]<CR><LF>
```

If the clearing of the error was not successful and the tracker is still in the error state:

```
CLR cmd_idx ERROR [Padding]<CR><LF>
CLR 01234 ERROR [Padding]<CR><LF>
```

7.87.11.5 pNotes

- cmd_idx will translate into a long, no units (sequence), with a range of 0 - 99999
- msg_idx will translate into a long, no units (sequence), with a range of 0 - 99999

7.87.12 MOVE_TIME

Usage:

```
MOVE_TIME  cmd_idx  pt_idx   time  X  Y  Z  rho theta phi
```

Sends a single, indexed time-based trajectory coordinate to the Tracker CS. This command will always be sent as two separate commands representing the two coordinates needed to calculate the total move time. The total move time represents the time to change from the current hardware state to either the Track or Slew mode and then move from the the current position to a position 1.5 seconds before the start of the trajectory. The coordinates (X, Y, Z, rho, theta, phi) are always specified as real tracker frame coordinates (RTF coordinate frame), which implies that the TCS has applied a mount model to take care of things like structure deformation. The index (cmd_idx) parameter has nothing to do with the trajectory coordinate, or even the position of a particular coordinate within a trajectory set. The tracker controller uses this index as part of its acknowledgement reply back to the TCS, giving the TCS confirmation that the tracker controller successfully received a particular coordinate. The point_index parameter (pt_idx) indicates the first or second coordinate. The time parameter is index time (number of seconds from an established 'zero point'). Index time will include any leap seconds that have occurred since the last crossing of the established zero point.

7.87.12.1 Reference

- cmd_idx = max 5 ascii bytes, followed by space (ASCII 32)
- pt_idx = max 1 ascii byte, followed by space (ASCII 32)
- time = 10 ascii bytes, followed by space (ASCII 32)
- X = max 9 ascii bytes, followed by space (ASCII 32)
- Y = max 9 ascii bytes, followed by space (ASCII 32)
- Z = max 8 ascii bytes, followed by space (ASCII 32)
- rho = max 7 ascii bytes, followed by space (ASCII 32)
- theta = max 7 ascii bytes, followed by space (ASCII 32)
- phi = max 7 ascii bytes, followed by space (ASCII 32)

7.87.12.2 Example

```
MOVE_TIME 01223 1 12345.0123 +234.654 -1200.987 123.455 -20.678 +2.0909 -4.5324
     [Padding]<CR><LF>
```

followed by:

```
MOVE_TIME 01224 2 12346.0123 +234.999 -1201.402 123.855 -20.012 +2.5990 -4.8324
     [Padding]<CR><LF>
```

7.87.12.3 Returns

The return response from the Tracker contains the command name, command index, the total move time and a status field. Note. The total move time is '0' for the first cooridinate received and if either command had errors.

Success:

```
MOVE_TIME 01223 0 OK [Padding]<CR><LF>
MOVE_TIME 01224 35 OK  [Padding]<CR><LF>
```

Error(s):

```
SET_TRAJ 01223 0 ERROR [Padding]<CR><LF>
SET_TRAJ 01224 0 ERROR [Padding]<CR><LF>
```

7.87.12.4 Notes

time will translate in to a double, and indicates the number of seconds from an established zero point cmd_idx will translate into a long, no units (sequence), with a range of 0 to 99999

- X will translate into a double, units are millimeters, with a range of +/- 1960.000 mm
- Y will translate into a double, units are millimeters, with a range of +/- 1940.000 mm
- Z will translate into a double, units are millimeters, with a range of -91.000 to +363.000 mm
- rho will translate into a double, units are degrees, with a range of +/- 27.000 deg
- theta will translate into a double, units are degrees, with a range of +/- 9.5000 deg
- phi will translate into a double, units are degrees, with a range of +/- 9.5000 deg

7.87.13 TMCS_return_values

- APITASK_STATUS_OK 0
- APITASK_STATUS_ERROR -1
- APITASK_COMMAND_ERROR -2
- APITASK_MOVE_ERROR -3
- APITASK_STATUS_COMPLETE 10

Classes

- class TMCSState

7.88 Tracker Server

7.88.1 Detailed Description

The Tracker Server condenses the TMCS (XXX need reference) API and makes the resulting commands available to all sub-systems.

Modules

- Tracker Motion Control System (TMCS) API

Chapter 8

Namespace Documentation

8.1 equ_hor Namespace Reference

8.1.1 Detailed Description

Equatorial - horizontal For going between (Alt, Az) and (RA, DEC) coordinate systems

Functions

- void equ2hor (const double ra, const double dec, const double lat, const double LST, double &az, double &alt)
- void **hor2equ** (const double az, const double alt, const double lat, const double LST, double &ra, double &dec)

8.1.2 Function Documentation

8.1.2.1 void equ_hor::equ2hor (const double *ra*, const double *dec*, const double *lat*, const double *LST*, double & *az*, double & *alt*)

Methods to go between equatorial (ra, dec) and horizontal (alt, az) All angles are in radians, as is LST. The zenith is not computed, but it is trivially obtained by zn = pi/2 - alt.

ra: right ascension dec: declination lat: observer's latitude LST: local sideral time az: azimuth alt: altitude (or elevation)

8.2 hetlib::hetutils Namespace Reference

8.2.1 Detailed Description

```
The hetutils.py package provides functions to do calculations specific
to the Hobby Eberly Telescope at McDonald Observatory. These functions
provide information about telescope pointing as well as tracker information.
```

Functions

- def degrees_to_hours
- def hours_to_degrees
- def radians_to_hours
- def hours_to_radians
- def modulo
- def clipToCircle_deg
- def clipToCircle_rad
- def BestAzimuth
- def Tde
- def Ho
- def Pa
- def Att
- def **space**

Variables

- float **Latitude_deg** = 30.681436
- tuple **Latitude_rad** = radians(Latitude_deg)
- float **LongitudeWest_deg** = 104.014742
- tuple **LongitudeWest_rad** = radians(LongitudeWest_deg)
- float **LongitudeEast_deg** = 360.0
- tuple **LongitudeEast_rad** = radians(LongitudeEast_deg)
- float **Altitude_deg** = 55.055223
- tuple **Altitude_rad** = radians(Altitude_deg)
- float **Height_meters** = 2003.0
- float **Fs_meters** = 13.37884180
- float **Fs_millimeters** = 1000.0
- tuple **het_P** = cos(Altitude_rad)
- tuple **het_Q** = sin(Altitude_rad)
- float **BetaLimit_deg** = 8.5
- tuple **BetaLimit_rad** = radians(BetaLimit_deg)
- float **X_offset_mm** = 3.030
- float **Y_offset_mm** = 25.690
- float **Z_offset_mm** = 0.025
- float **T_offset_mm** = 0.050
- float **P_offset_mm** = 0.100
- float **R_offset_mm** = 0.000
- float **TwoPi** = 2.0
- tuple **az_str** = str(az)
- tuple **t** = Tde(az)

- string **t_str** = '%8.4f'
- tuple **h** = Ho(az)
- string **h_str** = '%8.4f'
- tuple **p** = Pa(az)
- string **p_str** = '%8.4f'
- tuple **b** = BestAzimuth(dec)
- tuple **att** = Att(b, dec)
- string **b_str** = '%8.4f'
- tuple **dec_str** = str(dec)
- string **att_str** = '%8.4f'

8.2.2 Function Documentation

8.2.2.1 def hetlib::hetutils::Att (*az_deg*, *dec_deg*)

```
Calculate the total possible track time for a give azimuth and declination.
Input arguments are in degrees. The function returns the track time in minutes or
0 if no trajectory is available.
```

8.2.2.2 def hetlib::hetutils::BestAzimuth (*dec_deg*)

```
This function calculates the best azimuth for a given declination.
The function returns an azimuth only on the east side. If you want
the west azimuth, simply add 180 degrees to the return value.
The input declination is in degrees, The return value is the azimuth
in degrees or None is the declininination is not visible at HET.
```

8.2.2.3 def hetlib::hetutils::clipToCircle_deg (*x*)

```
Return the value (dX mod 360).  This has the result of clipping the
value between 0 <= dR < 360.0.

Inputs dX - the value you want clipped

Outputs none

Return (dX mod 360)
```

8.2.2.4 def hetlib::hetutils::clipToCircle_rad (*x*)

```
Return the value (dX mod 2PI).  This has the result of clipping the
value between 0 <= dR < 2PI.

Inputs dX - the value you want clipped

Outputs none

Return (dX mod 2PI)
```

8.2.2.5 def hetlib::hetutils::degrees_to_hours (*deg*)

```
Convert decimal degrees to decimal hours.
```

8.2.2.6 def hetlib::hetutils::Ho (*azimuth_deg*)

```
Calculate the telecentric hour angle for a given azimuth.
Negative values are east hour angles.  The input azimuth is in
degrees. Returns the hour angle in degrees.
```

8.2.2.7 def hetlib::hetutils::hours_to_degrees (*hr*)

```
Convert decimal hours to decimal degrees.
```

8.2.2.8 def hetlib::hetutils::hours_to_radians (*hr*)

```
Convert decimal hours to decimal radians.
```

8.2.2.9 def hetlib::hetutils::modulo (*x*, *y*)

```
A general purpose modulo (x mod y) function.

Definition of 'mod'
x mod y = x - y * floor( x/y)  for y != 0
x mod 0 = x

if 0 < y then 0 <= (x mod y) < y
if 0 > y then 0 >= (x mod y) > y
```

8.2.2.10 def hetlib::hetutils::Pa (*azimuth_deg*)

```
Calculate the zero-crossing paralactic angle as a function of azimuth.
The input azimuth is in degrees. Returns the paralactic angle in degrees.
```

8.2.2.11 def hetlib::hetutils::radians_to_hours (*rad*)

```
Convert decimal radians to decimal hours.
```

8.2.2.12 def hetlib::hetutils::Tde (*azimuth_deg*)

```
Calculate the telecentric declination as a function of azimuth.
The input azimuth is in degrees. Returns the declination in degrees
or None if the input is not between 0 <= az < 360.0
```

8.3 hetlib::Points Namespace Reference

8.3.1 Detailed Description

```
This file contains the list ``Points`` which consists of of 97 dictionary defining
points in ITF coordinates that lie on the focal sphere.  The dictionarys are defined as

.. code-block:: python

  Points = [
  {'Index':  1,
   'X':  790.877147,
   'Y': 1812.127196,
   'Z':  146.930811,
   'Theta': 7.799501970,
   'Phi':   3.421093209,
   'Rho':   0.000000000,
   'Beta':  8.500000000},
   ...
   ...
   ...
   {'Index': 97,
    'X': -790.877147,
    'Y':-1812.127196,
    'Z':  146.930811,
    'Theta': -7.799501970,
    'Phi':   -3.421093209,
    'Rho':    0.000000000, '
    'Beta':   8.500000000},
   ]


The parameters ``X``, ``Y``, and ``Z`` are given in millimeter while
``Theta``, ``Phi``, ``Rho``, and ``Beta`` are given in units of
degrees

These points are primarily used in the program mountTest (DEFINE LINK
HERE) but can be used anywhere.  The points were calculated using a
tracker sphere of ????? mm.
```

Functions

- def **plot_list**

Variables

- list **Points**
- dictionary **knownLists**

8.3.2 Variable Documentation

8.3.2.1 dictionary hetlib::Points::knownLists

Initial value:

```
{
    'List_97' : range(1,98),
    'List_21' : [49, 47, 45, 23, 7, 25, 27, 9, 11, 29, 31, 53, 51, 73, 75, 91, 89
      , 87, 67, 69, 71, 49],
```

```
    'List_21t' : [7, 9, 11, 23, 25, 27, 29, 31, 45, 47, 49, 51, 53, 67, 69, 71, 7
      3, 75, 87, 89, 91],
    'List_21a' : [27, 9, 11, 29, 32, 53, 51, 73, 75, 91, 89, 87, 67, 69, 71, 49],

    'List_21r' : [49, 27, 9, 11, 25, 67, 69, 71, 75, 91, 89, 87, 29, 32, 53, 51,
      73, 47, 45, 23, 7, 49],
    'List_2' : [49, 89],
    'List_2r' : [89, 49],
    'List_05' : [9, 8, 10, 3, 2],
    'List_Q1' : [29, 1, 2, 3, 9, 10, 11, 12, 13, 14, 15, 16, 17, 27, 28, 29, 30,
      31, 32, 33, 34, 35, 36, 37, 38, 49, 50, 51, 52, 53, 54],
    'List_Q2' : [73, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 71, 72, 73,
      74, 75, 76, 77, 78, 79, 80, 81, 89, 90, 91, 92, 93, 94, 95],
    'List_Q3' : [69, 44, 45, 46, 47, 48, 49, 60, 61, 62, 63, 64, 65, 66, 67, 68,
      69, 70, 71, 81, 82, 83, 84, 85, 86, 87, 88, 89, 95, 96, 97],
    'List_Q4' : [25, 3, 4, 5, 6, 7, 8, 9, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,
       27, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]
}
```

8.4 hetlib::tcsconfig Namespace Reference

8.4.1 Detailed Description

```
``tcsconfig.py`` defines the standard IP address and port numbers
for the routes and event routes of the control systems.  It also
builds the URI strings and defines the basic command strings to start
the control systems. These command strings contain the route and event
route definition for the control systems as well the location of the
configuration files. The environment variables **HET_SRC_ROOT** and
**HET_DEPLOY_DIR** are read to find the location of the executables. The
default location of the configuration files can be changed by setting
the environment variable **HET_CONF**. The location of the log
files is defined in the configuration files.  Currently the control
systems run out of **HET_SRC_ROOT** but this will be changed to
**HET_DEPLOY_DIR** sometime soon.
```

Variables

- list **namedIP** = os.environ['HET_NAMED_IP']
- string **log_relayIP** = '192.168.66.99'
- string **named_route_port** = '30000'
- string **named_event_route_port** = '30001'
- string **log_relay_route_port** = '31000'
- string **log_relay_event_route_port** = '31001'
- string **named_route** = 'tcp://'
- string **named_event_route** = 'tcp://'
- string **log_relay_route** = 'tcp://'
- string **log_relay_event_route** = 'tcp://'

8.5 hetlib::tcsutils Namespace Reference

8.5.1 Detailed Description

```
tcsutils.py
Provides useful tools (or maybe not)
```

Functions

- def **kill_all_subs**
- def **kill_remaining_subs**
- def start_clients
- def **dummy**

Variables

- list **namedIP** = os.environ['HET_NAMED_IP']
- string **log_relayIP** = '192.168.66.99'
- string **named_route_port** = '30000'
- string **named_event_route_port** = '30001'
- string **log_relay_route_port** = '31000'
- string **log_relay_event_route_port** = '31001'
- string **named_route** = 'tcp://'
- string **named_event_route** = 'tcp://'
- string **log_relay_route** = 'tcp://'
- string **log_relay_event_route** = 'tcp://'
- list **het_deploy_dir** = os.environ['HET_DEPLOY_DIR']
- tuple **tnmd** = TCSNamed.TCSNamed(named_route)

8.5.2 Function Documentation

8.5.2.1 def hetlib::tcsutils::start_clients (*tcs* = `False`, *tracker* = `False`, *legacy* = `False`, *pas* = `False`, *pfip* = `False`, *apc* = `False`, *lrs2* = `False`, *virus* = `False`, *vdas* = `False`, *named* = `False`, *route* = `''`, *verbose* = `False`)

```
Start client communications as requested.  Returns a dictionary with the
client handles.  May be called multiple times with different arguments.
```

8.6 hetutils Namespace Reference

8.6.1 Detailed Description

```
The hetutils.py package provides functions to do calculations specific
to the Hobby Eberly Telescope at McDonald Observatory. These functions
provide information about telescope pointing as well as tracker information.

Note these equations are based on Frank Ray's TR-43 report and may not accurately
reflect that existing tracker. In particular, these equations do not include
the mount models. However,the equations should be good to at least a
millimeter.
```

Functions

- def degrees_to_hours
- def hours_to_degrees
- def radians_to_hours
- def hours_to_radians
- def modulo
- def clipToCircle_deg
- def clipToCircle_rad
- def BestAzimuth
- def Tde
- def Ho
- def Pa
- def Att
- def **space**

Variables

- float **Latitude_deg** = 30.681436
- tuple **Latitude_rad** = radians(Latitude_deg)
- float **LongitudeWest_deg** = 104.014742
- tuple **LongitudeWest_rad** = radians(LongitudeWest_deg)
- float **LongitudeEast_deg** = 360.0
- tuple **LongitudeEast_rad** = radians(LongitudeEast_deg)
- float **Altitude_deg** = 55.0
- tuple **Altitude_rad** = radians(Altitude_deg)
- float **Height_meters** = 2003.0
- float **Fs_meters** = 13.37884180
- float **Fs_millimeters** = 1000.0
- tuple **het_P** = cos(Altitude_rad)
- tuple **het_Q** = sin(Altitude_rad)
- float **BetaLimit_deg** = 8.5
- tuple **BetaLimit_rad** = radians(BetaLimit_deg)
- float **X_offset_mm** = 3.030
- float **Y_offset_mm** = 25.690
- float **Z_offset_mm** = 0.025
- float **T_offset_mm** = 0.050
- float **P_offset_mm** = 0.100

- float **R_offset_mm** = 0.000
- float **TwoPi** = 2.0
- tuple **az_str** = str(az)
- tuple **t** = Tde(az)
- string **t_str** = '%8.4f'
- tuple **h** = Ho(az)
- string **h_str** = '%8.4f'
- tuple **p** = Pa(az)
- string **p_str** = '%8.4f'
- tuple **b** = BestAzimuth(dec)
- tuple **att** = Att(b, dec)
- string **b_str** = '%8.4f'
- tuple **dec_str** = str(dec)
- string **att_str** = '%8.4f'

8.6.2 Function Documentation

8.6.2.1 def hetutils::Att (*az_deg*, *dec_deg*)

```
Calculate the total possible track time for a give azimuth and declination.
Input arguments are in degrees. The function returns the track time in minutes or
0 if no trajectory is available.
```

8.6.2.2 def hetutils::BestAzimuth (*dec_deg*)

```
This function calculates the best azimuth for a given declination.
The function returns an azimuth only on the east side. If you want
the west azimuth, simply add 180 degrees to the return value.
The input declination is in degrees, The return value is the azimuth
in degrees or None is the declinination is not visible at HET.
```

8.6.2.3 def hetutils::clipToCircle_deg (*x*)

```
Return the value (dX mod 360).  This has the result of clipping the
value between 0 <= dR < 360.0.

Inputs dX - the value you want clipped

Outputs none

Return (dX mod 360)
```

8.6.2.4 def hetutils::clipToCircle_rad (*x*)

```
Return the value (dX mod 2PI).  This has the result of clipping the
value between 0 <= dR < 2PI.

Inputs dX - the value you want clipped

Outputs none

Return (dX mod 2PI)
```

8.6.2.5 def hetutils::degrees_to_hours (*deg*)

```
Convert decimal degrees to decimal hours.
```

8.6.2.6 def hetutils::Ho (*azimuth_deg*)

```
Calculate the telecentric hour angle for a given azimuth.
Negative values are east hour angles.  The input azimuth is in
degrees. Returns the hour angle in degrees.
```

8.6.2.7 def hetutils::hours_to_degrees (*hr*)

```
Convert decimal hours to decimal degrees.
```

8.6.2.8 def hetutils::hours_to_radians (*hr*)

```
Convert decimal hours to decimal radians.
```

8.6.2.9 def hetutils::modulo (*x*, *y*)

```
A general purpose modulo (x mod y) function.

Definition of 'mod'
x mod y = x - y * floor( x/y)  for y != 0
x mod 0 = x

if 0 < y then 0 <= (x mod y) < y
if 0 > y then 0 >= (x mod y) > y
```

8.6.2.10 def hetutils::Pa (*azimuth_deg*)

```
Calculate the zero-crossing paralactic angle as a function of azimuth.
The input azimuth is in degrees. Returns the paralactic angle in degrees.
```

8.6.2.11 def hetutils::radians_to_hours (*rad*)

```
Convert decimal radians to decimal hours.
```

8.6.2.12 def hetutils::Tde (*azimuth_deg*)

```
Calculate the telecentric declination as a function of azimuth.
The input azimuth is in degrees. Returns the declination in degrees
or None if the input is not between 0 <= az < 360.0
```

8.7 Points Namespace Reference

8.7.1 Detailed Description

```
This file contains the list ``Points`` which consists of of 97 dictionary defining
points in ITF coordinates that lie on the focal sphere.  The dictionarys are defined as

.. code-block:: python

  Points = [
  {'Index':  1,
   'X':  790.877147,
   'Y': 1812.127196,
   'Z':  146.930811,
   'Theta': 7.799501970,
   'Phi':   3.421093209,
   'Rho':   0.000000000,
   'Beta':  8.500000000},
   ...
   ...
   ...
   {'Index': 97,
    'X': -790.877147,
    'Y':-1812.127196,
    'Z':  146.930811,
    'Theta': -7.799501970,
    'Phi':   -3.421093209,
    'Rho':    0.000000000, '
    'Beta':   8.500000000},
   ]


The parameters ``X``, ``Y``, and ``Z`` are given in millimeter while
``Theta``, ``Phi``, ``Rho``, and ``Beta`` are given in units of
degrees

These points are primarily used in the program mountTest (DEFINE LINK
HERE) but can be used anywhere.  The points were calculated using a
tracker sphere of ????? mm.
```

Functions

- def **plot_list**

Variables

- list **Points**
- dictionary **knownLists**

8.7.2 Variable Documentation

8.7.2.1 dictionary Points::knownLists

Initial value:

```
{
    'List_97' : range(1,98),
    'List_21a' : [7, 9, 11, 23, 25, 27, 29, 31, 45, 47, 49, 51, 53, 67, 69, 71, 7
      3, 75, 87, 89, 91],
```

```
    'List_21' : [49, 47, 45, 23, 7, 25, 27, 9, 11, 29, 32, 53, 51, 73, 75, 91, 89
      , 87, 67, 69, 71, 49],
    'List_21a' : [27, 9, 11, 29, 32, 53, 51, 73, 75, 91, 89, 87, 67, 69, 71, 49],

    'List_21r' : [49, 27, 9, 11, 25, 67, 69, 71, 75, 91, 89, 87, 29, 32, 53, 51,
      73, 47, 45, 23, 7, 49],
    'List_2' : [49, 89],
    'List_2r' : [89, 49],
    'List_05' : [9, 8, 10, 3, 2],
    'List_Q1' : [29, 1, 2, 3, 9, 10, 11, 12, 13, 14, 15, 16, 17, 27, 28, 29, 30,
      31, 32, 33, 34, 35, 36, 37, 38, 49, 50, 51, 52, 53, 54],
    'List_Q2' : [73, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 71, 72, 73,
      74, 75, 76, 77, 78, 79, 80, 81, 89, 90, 91, 92, 93, 94, 95],
    'List_Q3' : [69, 44, 45, 46, 47, 48, 49, 60, 61, 62, 63, 64, 65, 66, 67, 68,
      69, 70, 71, 81, 82, 83, 84, 85, 86, 87, 88, 89, 95, 96, 97],
    'List_Q4' : [25, 3, 4, 5, 6, 7, 8, 9, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,
       27, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49]
}
```

8.8 pt_itf Namespace Reference

8.8.1 Detailed Description

Pointing - sky For going between telescope pointing and (RA, DEC)

Functions

- bool **pt2itf** (const double RA, const double DEC, const altaz_coordinate &tel, const double UTCasMJD, const double dUT1, const double x_polar, const double y_polar, const double temperature, const double pressure, const double relhum, itf_coordinate &itf, double &beta, double &tde, double &pa, double &h, double &hc, double r, double &d, rho_derivation rd, vec3_t &rho_ref_vec)
- void itf2pt (const altaz_coordinate &tel, const itf_coordinate &itf, const double dUT1, const double x_polar, const double y_polar, const double temperature, const double pressure, const double relhum, double *RA, double *DEC)

  convert the current ITF value to current apparent RA, DEC in radians.

- itf_coordinate deltasky2itf (const double d_ang_x, const double d_ang_y, const itf_coordinate &itf)

  This routine converts a displacement in telecentric x and y angles to a displacement in ITF relative to the current values.

- itf_coordinate deltafocus2itf (const double dw, const itf_coordinate &itf)

  convert a displacement in the w (focus) axis to a displacement in ITF relative to the current ITF value

- void find_radec (double x, double y, double pa, double hc, double tde, double MJD, double &obs_ra, double &obs_dec)
- bool inEllipse (const double x, const double y, const double beta, const double theta, const double phi)

8.8.2 Function Documentation

8.8.2.1 void pt_itf::find_radec (double *x*, double *y*, double *pa*, double *hc*, double *tde*, double *MJD*, double & *obs_ra*, double & *obs_dec*)

find_radec - find the observed RA and Dec for the input tracker X,Y

Inputs (all doubles): tracker X position in mm tracker Y position in mm parallactic angle in radians telecentric hour angle in radians telecentric declination in radians MJD(UTC) in days dUT1 in seconds

Outputs (all doubles): observed RA in radians observed Dec in radians

Notes:

1) Be careful of UT1 vs. UTC vs. TT. It might be better to define MJD to always be UT1 or UTC, but we need both. It might be better to just label the MJD variables so it is always clear which is which.

2) See the discussion in ha_to_utc.notes for details and references.

09 JAN 2011 Modified by M. E. Cornell from nsotraj.c dated 07JAN11, with ideas from the HET TCS routine rdscope. 11 JAN 2011 Note that this routine takes UTC as an MJD. Added dUT1 as an argument. 20 MAR 2011 Add a check at the end for negative RA. AUG 2013 Integrated in point library by Martin Landriau. AUG 2015 Moved to pt_itf by Martin Landriau

8.8.2.2 bool pt_itf::inEllipse (const double *x*, const double *y*, const double *beta*, const double *theta*, const double *phi*)

Condition for which a point along an object's trajectory which is inside the HET's range. Used to determine start and end time of the object's HET trajectory and to check that a given pointing is on sphere.

8.9 st3 Namespace Reference

8.9.1 Detailed Description

```
Read SET_TRAJ commands from TMCS 'listen log' send over socket to Tracker
```

Functions

- def **main**
- def getCurrentIndexTime

Variables

- **hrwd** = True;

8.9.2 Function Documentation

8.9.2.1 def st3::getCurrentIndexTime ()

```
Convert datetime.datetime() to an index time. Note that we loss
the date information in the datetime() object.

input; datetime.datetime()
output: float
```

8.10 tcsconfig Namespace Reference

8.10.1 Detailed Description

```
``tcsconfig.py`` read the environment variables **HET_SRC_ROOT** and
**HET_DEPLOY_DIR** to find the location of the executables. The
default location of the configuration files can be changed by setting
the environment variable **HET_CONF**. The location of the log
files is defined in the configuration files.  Currently the control
systems run out of **HET_DEPLOY_DIR**.

This file may be deprecated in the future. Currently it does very litte
and should probably not be imported directly into python programs.
```

Variables

- string **tcsconfig_version** = 'tcsconfig.py v1.0 dtd 7 Aug 2014'
- list **src_root** = os.environ['HET_SRC_ROOT']
- list **het_deploy_dir** = os.environ['HET_DEPLOY_DIR']
- string **namedIP** = '192.168.66.99'
- string **named_route_port** = '30000'
- string **named_event_route_port** = '30001'
- string **named_route** = 'tcp://'
- string **named_event_route** = 'tcp://'

8.11 tcsutils Namespace Reference

8.11.1 Detailed Description

```
tcsutils.py
Provides useful tools (or maybe not)
```

Functions

- def **kill_all_subs**
- def **kill_remaining_subs**
- def named_lookup
- def start_clients
- def **printVerbose**
- def **printErr**
- def getposition
- def moveStructure
- def **dummy**

8.11.2 Function Documentation

8.11.2.1 def tcsutils::getposition (*tcs* = `None`)

```
Use the handler tcs.tracker_position() to get the current
position of the tracker.  If the argument tcs is not None, then
we use it as the client handle, otherwise we try to open a client
ourselves.
```

8.11.2.2 def tcsutils::moveStructure (*az_deg*, *legacy*)

```
Send a structure motion command to move the structure to
az_deg position through the control system client legacy.

Return az, el
```

8.11.2.3 def tcsutils::named_lookup (*key* = `None`)

```
Contact tcsnamed server and lookup a key.  Returns the value
found. We return None or if there is no such key. If we can not contact the tcsnamed server,
then an error will be raised by TCSSubSystem().

We use a random number in the name string because the return object is not
fully deleted with the ``del tcsnmd``.  This appears to be an issue with the python
garbage collection not being implemented quickly.  For processes that make a large
number of calls to start_client() or named_lookup(), this will cause a memeory leak.
I have inplemented the random number in order to get the system working again quickly.
However, future work should be aware of this problem.
```

8.11.2.4 def tcsutils::start_clients (*tcs* = `False`, *tracker* = `False`, *legacy* = `False`, *pas* = `False`, *pfip* = `False`, *apc* = `False`, *lrs2* = `False`, *vdas* = `False`, *named* = `False`, *verbose* = `False`)

```
Start client communications as requested.  Returns a dictionary with the
client handles.  May be called multiple times with different arguments.
```

8.12 tthread Namespace Reference

8.12.1 Detailed Description

Main name space for TinyThread++. This namespace is more or less equivalent to the `std` namespace for the C++11 thread classes. For instance, the tthread::mutex class corresponds to the std::mutex class.

Namespaces

- namespace chrono
- namespace this_thread

Classes

- class fast_mutex
- class ThreadPoolException
- struct ThreadPoolWorker
- class ThreadPool
- class Timer
- class ThreadWorker
- class mutex
- class recursive_mutex
- class lock_guard
- class condition_variable
- struct _thread_start_info
- struct _thread_start_functor
- struct _thread_start_functor_arg
- struct _thread_start_member
- struct _thread_start_member_arg
- class thread

  Thread class.

- class ratio

Typedefs

- typedef void *(* pthread_trampoline)(void *)
- typedef long long **__intmax_t**

Functions

- void * _threadpool_start_thread (void *arg)

8.12.2 Typedef Documentation

8.12.2.1 typedef void*(* tthread::pthread_trampoline)(void *)

Base class and derived classes for thread trampolines to allow threading of

- free functions
- free functions with one argument
- class member functions
- class memeber functions with one argument

These are used in `thread`, `ThreadWorker`, and `ThreadPool`

8.12.3 Function Documentation

8.12.3.1 void* tthread::_threadpool_start_thread (void * *arg*)

trampoline function for pthreads, calls `ThreadPool::execute_thread()`

8.13 tthread::chrono Namespace Reference

8.13.1 Detailed Description

Minimal implementation of the `chrono` namespace. The `chrono` namespace provides types for specifying time intervals.

Classes

- class duration

Typedefs

- typedef duration< __intmax_t, ratio< 1, 1000000000 > > nanoseconds

  Duration with the unit nanoseconds.

- typedef duration< __intmax_t, ratio< 1, 1000000 > > microseconds

  Duration with the unit microseconds.

- typedef duration< __intmax_t, ratio< 1, 1000 > > milliseconds

  Duration with the unit milliseconds.

- typedef duration< __intmax_t > seconds

  Duration with the unit seconds.

- typedef duration< __intmax_t, ratio< 60 > > minutes

  Duration with the unit minutes.

- typedef duration< __intmax_t, ratio< 3600 > > hours

  Duration with the unit hours.

8.14 tthread::this_thread Namespace Reference

8.14.1 Detailed Description

The namespace `this_thread` provides methods for dealing with the calling thread.

Functions

- thread::id get_id ()

  Return the thread ID of the calling thread.

- void yield ()
- template<class _Rep , class _Period >
  void sleep_for (const chrono::duration< _Rep, _Period > &aTime)

8.14.2 Function Documentation

8.14.2.1 template<class _Rep , class _Period > void tthread::this_thread::sleep_for (const chrono::duration< _Rep, _Period > & *aTime*) [inline]

Blocks the calling thread for a period of time.

Parameters:

← ***aTime*** Minimum time to put the thread to sleep. Example usage:

```
// Sleep for 100 milliseconds
this_thread::sleep_for(chrono::milliseconds(100));
```

Note:

Supported duration types are: nanoseconds, microseconds, milliseconds, seconds, minutes and hours.

8.14.2.2 void tthread::this_thread::yield () [inline]

Yield execution to another thread. Offers the operating system the opportunity to schedule another thread that is ready to run on the current processor.

Chapter 9

Class Documentation

9.1 _modbus Struct Reference

Public Attributes

- int **slave**
- int **s**
- int **debug**
- int **error_recovery**
- struct timeval **response_timeout**
- struct timeval **byte_timeout**
- const modbus_backend_t * **backend**
- void * **backend_data**

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

- pfip/MODBUS_lib/include/modbus-private.h

9.2 _modbus_backend Struct Reference

Public Attributes

- unsigned int **backend_type**
- unsigned int **header_length**
- unsigned int **checksum_length**
- unsigned int **max_adu_length**
- int(∗ **set_slave**)(modbus_t ∗ctx, int slave)
- int(∗ **build_request_basis**)(modbus_t ∗ctx, int function, int addr, int nb, uint8_t ∗req)
- int(∗ **build_response_basis**)(sft_t ∗sft, uint8_t ∗rsp)
- int(∗ **prepare_response_tid**)(const uint8_t ∗req, int ∗req_length)
- int(∗ **send_msg_pre**)(uint8_t ∗req, int req_length)
- ssize_t(∗ **send**)(modbus_t ∗ctx, const uint8_t ∗req, int req_length)
- ssize_t(∗ **recv**)(modbus_t ∗ctx, uint8_t ∗rsp, int rsp_length)
- int(∗ **check_integrity**)(modbus_t ∗ctx, uint8_t ∗msg, const int msg_length)
- int(∗ **pre_check_confirmation**)(modbus_t ∗ctx, const uint8_t ∗req, const uint8_t ∗rsp, int rsp_-length)
- int(∗ **connect**)(modbus_t ∗ctx)
- void(∗ **close**)(modbus_t ∗ctx)
- int(∗ **flush**)(modbus_t ∗ctx)
- int(∗ **select**)(modbus_t ∗ctx, fd_set ∗rfds, struct timeval ∗tv, int msg_length)
- int(∗ **filter_request**)(modbus_t ∗ctx, int slave)

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

- pfip/MODBUS_lib/include/modbus-private.h

9.3 _modbus_rtu Struct Reference

Public Attributes

- char **device** [16]
- int **baud**
- uint8_t **data_bit**
- uint8_t **stop_bit**
- char **parity**
- struct termios **old_tios**
- int **serial_mode**

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

- pfip/MODBUS_lib/include/modbus-rtu-private.h

9.4 _modbus_tcp Struct Reference

Public Attributes

- int **port**
- char **ip** [16]

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

- pfip/MODBUS_lib/include/modbus-tcp-private.h

9.5 _modbus_tcp_pi Struct Reference

Public Attributes

- int **port**
- char **node** [1025]
- char **service** [32]

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

- pfip/MODBUS_lib/include/modbus-tcp-private.h

9.6 _sft Struct Reference

Public Attributes

- int **slave**
- int **function**
- int **t_id**

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

- pfip/MODBUS_lib/include/modbus-private.h

9.7 tthread::_thread_start_functor< T > Struct Template Reference

template<typename T> struct tthread::_thread_start_functor< T >

Public Member Functions

- **_thread_start_functor** (T functor)
- virtual void **run** ()

Public Attributes

- T **functor_**

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

- het_lib/tthread/include/tinythread.h

9.8 tthread::_thread_start_functor_arg< F, A > Struct Template Reference

template<typename F, typename A> struct tthread::_thread_start_functor_arg< F, A >

Public Member Functions

- **_thread_start_functor_arg** (F function, A arg)
- virtual void **run** ()

Public Attributes

- F **function_**
- A **_arg**

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

- het_lib/tthread/include/tinythread.h

9.9 tthread::_thread_start_info Struct Reference

Public Member Functions

- virtual void **run** ()=0

Public Attributes

- thread * **thread_**
- pthread_trampoline **trampoline_**

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

- het_lib/tthread/include/tinythread.h

9.10 tthread::_thread_start_member< 14463331 > Struct Template Reference

template<typename 173. 14463331> struct tthread::_thread_start_member< 14463331 >

Public Member Functions

- **_thread_start_member** (173.14463331 *object, void(173.14463331::*function)())
- virtual void **run** ()
- **void** (173.14463331::*function_)()

Public Attributes

- * **object_**

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

- het_lib/tthread/include/tinythread.h

9.11 tthread::_thread_start_member_arg< 14463331, A > Struct Template Reference

template<typename 173. 14463331, typename A> struct tthread::_thread_start_member_arg< 14463331, A >

Public Member Functions

- **_thread_start_member_arg** (173.14463331 *object, void(173.14463331::*function)(A), A a)
- virtual void **run** ()
- **void** (173.14463331::*function_)(A)

Public Attributes

- * **object_**
- A **arg_**

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

- het_lib/tthread/include/tinythread.h

9.12 Abort Class Reference

Abort current Dome command (normally movement).

9.12.1 Detailed Description

Abort the current shutter command.

Parameters:

none

`#include <legacyDomeSubSystem.h>`

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

- legacy/include/legacyDomeSubSystem.h

9.13 camra_handlers::abort Class Reference

Public Member Functions

- **abort** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.13.1 Member Function Documentation

9.13.1.1 void camra_handlers::abort::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.14 abort Class Reference

9.14.1 Detailed Description

Aborts the currenlty active exposure discarding all data.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.15 abortMove Class Reference

9.15.1 Detailed Description

Aborts running trajectory by sending ABORT command to TMCS.

Optional parameters:

id<int> - The ID of the trajectory to abort. Defaults to 0

```
#include <tracker_handler_doc.h>
```

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

- tracker/tracker_handler_doc.h

9.16 tracker_handlers::abortMove Class Reference

Public Member Functions

- **abortMove** (TrackerDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.16.1 Member Function Documentation

9.16.1.1 void tracker_handlers::abortMove::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tracker/tracker_handlers.h

9.17 Ui::aboutDialog Class Reference

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

- gui/build/ui_aboutDlg.h

9.18 aboutDialog Class Reference

Public Member Functions

- **aboutDialog** (QWidget *parent=0)
- **aboutDialog** (QWidget *parent=0)

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

- gui/include/mainWindow.h
- utility_gui/include/mainWindow.h

9.19 ACQ_Camera Class Reference

A class to represent the acquisition camera.

`#include <ACQCamera.h>`

Public Member Functions

- ACQ_Camera ()
  Constructor.
- bool Initialize (EPICSComm *const epicsComm)
  Initialize the camera object with hardware encoding values.
- bool Enable (EPICSComm *const epicsComm)
  Command the acquisition camera to become enabled.
- void SetACQCameraStatus (MAXONDriveStatus &status)
  Store the hardware status bit values in the camera object.
- void GetACQCameraStatus (MAXONDriveStatus &status)
  Retrieve the hardware status bit values from the camera object.
- bool SetACQCameraFilter (EPICSComm *const epicsComm, const int filterPos)
  Change the acquisition camera filter setting.
- pfip::ACQCameraFilterPos GetACQCameraFilterPos ()
  Return the hardware value for the acquisition camera filter setting.
- pfip::MirrorSetting GetACQCameraMirrorPos ()
  Return the acquisition camera mirror position (retracted/deployed).
- void SetACQCameraMirrorPos (pfip::MirrorSetting pos)
  Set the acquisition camera mirror position.
- bool SetACQCameraPower (EPICSComm *epicsComm, pfip::PowerSetting power)
  Turn the acquisition camera mirror power on/off.
- pfip::PowerSetting GetACQCameraPower ()
  Get power on/off for acquisition camera.
- void SetACQCameraFocus (int setting)
  Change the acquisition camera focus by some amount.
- int GetACQCameraFocus ()
  Get the acquisition camera focus setting.
- bool ReadCameraStatus (EPICSComm *const epicsComm)
  Read the camera hardware status bits.

Static Public Member Functions

- static int **FilterPosToActual** (const pfip::ACQCameraFilterPos filterPos)
- static pfip::ACQCameraFilterPos **ActualToFilterPos** (int actualFilterPos)

9.19.1 Member Function Documentation

9.19.1.1 bool ACQ_Camera::Enable (EPICSComm ∗const *epicsComm*)

Parameters:

epicsCon[in] - ptr to EPICS communication object

9.19.1.2 pfip::ACQCameraFilterPos ACQ_Camera::GetACQCameraFilterPos () `[inline]`

Parameters:

epicsComm[in] - ptr to EPICS communication object Return the hardware value for the acquisition camera filter setting from camera object

9.19.1.3 void ACQ_Camera::GetACQCameraStatus (MAXONDriveStatus & *status*)

Parameters:

status[out] - ptr to location to store the hardware status bits

9.19.1.4 bool ACQ_Camera::Initialize (EPICSComm ∗const *epicsComm*)

Parameters:

epicsComm[in] - ptr to EPICS communication object

9.19.1.5 bool ACQ_Camera::SetACQCameraFilter (EPICSComm ∗const *epicsComm*, const int *filterPos*)

Parameters:

epicsComm[in] - ptr to the EPICS communication object

pos[in] - the new camera filter setting

9.19.1.6 void ACQ_Camera::SetACQCameraStatus (MAXONDriveStatus & *status*)

Parameters:

status[in] - ptr to structure containing hardware status bits

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

- pfip/include/ACQCamera.h
- pfip/src/ACQCamera.cpp

9.20 ACQ_Shutter Class Reference

A class to represent the acquisition camera shutter.

`#include <ACQShutter.h>`

Public Types

- enum **shutterHandler** {
  initShutter, **openShutter**, **closeShutter**, **statusHandler**,
  otherHandler }

Public Member Functions

- ACQ_Shutter ()
  Constructor.
- virtual ~ACQ_Shutter ()
  Destructor.
- bool Initialize (EPICSComm *const epicsComm)
  Initialize the shutter object with values from currend hardware encoder settings.
- bool OpenShutter (EPICSComm *const epicsComm)
  Open the acquisition system shutter.
- bool CloseShutter (EPICSComm *const epicsComm)
  Close the acquisition system shutter.
- bool HomeShutter (EPICSComm *const epicsComm)
  Home the acquisition system shutter.
- bool EnableShutter (EPICSComm *const epicsComm)
  Enable the shutter for activity.
- bool IsShutterClosed ()
  Returns true if shutter is closed.
- bool **IsShutterOpen** ()
- void SetShutterStatus (MAXONDriveStatus &status)
  Store the shutter status flags in the shutter object.
- void GetShutterStatus (MAXONDriveStatus &status)
  Retrieve the hardware status flags from the shutter object.
- bool WriteShutterOpenClose (pfip::EPICSVariableLookupKey varKey, EPICSComm *const epicsComm)
  Send a new encoder setting for the shutter position.

- bool **WriteShutterSetting** (const uint val, EPICSComm *const epicsComm)
- bool **GetShutterSetting** (uint *val, EPICSComm *const epicsComm)
- uint GetRawSetting ()

  Retrieve the encoder setting from the shutter object.

- void SetEncoderInternalRawValue (const uint rawValue)

  Return the shutter object's raw encoder setting.

- pfip::FPAShutterState GetInternalState ()

  Return the shutter state from the shutter object.

- void SetInternalState (pfip::FPAShutterState state)

  Change the shutter state value in the shutter object.

- bool ExposureSetup ()

  Wait for all hardware to arrive in the proper state for an exposure.

- bool WaitForShutterClose (EPICSComm *epicsComm, const fptime &timeOut=fptime(0, 0))

  Wait until the shutter reaches the closed or open position.

- bool **WaitForShutterOpen** (EPICSComm *epicsComm)
- void **SetLastOpened** (double openTime)
- void SetLastClosed (double lastCloseTime)

  Set the value for last time shutter was closed.

- fptime & GetLastOpenTime ()

  Get time the shutter last opened.

- fptime & GetLastCloseTime ()

  Get time the shutter last closed.

- bool ReadShutterStatus (EPICSComm *epicsComm, shutterHandler handler=otherHandler)

  Read hardware status bits for shutter.

- void InterruptRequest ()

  Interrupt a shutter operation.

- bool **WriteShutterTimes** (fptime &start, fptime &end, EPICSComm *const epicsComm)
- bool HandleExposure (const fptime &duration, EPICSComm *const epicsComm)

  Open the shutter for a specified duration, then close.

- void SetSchedOpen (fptime &openSchedule)

  set operators for the schedule open/close time

- void **SetSchedClose** (fptime &closeSchedule)
- fptime & **GetScheduleOpenTime** ()
- fptime & **GetScheduleCloseTime** ()
- fptime & **GetShutterOpenDuration** ()

9.20.1 Constructor & Destructor Documentation

9.20.1.1 ACQ_Shutter::ACQ_Shutter ()

Parameters:

← - ptr to PFIP configuration object

9.20.2 Member Function Documentation

9.20.2.1 bool ACQ_Shutter::CloseShutter (EPICSComm ∗const *epicsComm*)

Parameters:

epicsComm[in] - ptr to the epics communication object

9.20.2.2 bool ACQ_Shutter::EnableShutter (EPICSComm ∗const *epicsComm*)

Parameters:

epicsComm[in] - ptre to the epics communication object

9.20.2.3 void ACQ_Shutter::GetShutterStatus (MAXONDriveStatus & *status*) `[inline]`

Parameters:

status[out] - ptr to place to store hardware status bits

9.20.2.4 bool ACQ_Shutter::HomeShutter (EPICSComm ∗const *epicsComm*)

Parameters:

epicsComm[in] - ptr to the epics communication object

9.20.2.5 bool ACQ_Shutter::Initialize (EPICSComm ∗const *epicsComm*)

Parameters:

epicsComm[in] - ptr to the epics communication object

9.20.2.6 bool ACQ_Shutter::OpenShutter (EPICSComm ∗const *epicsComm*)

Parameters:

epicsComm[in] - ptr to the epics communication object

9.20.2.7 void ACQ_Shutter::SetShutterStatus (MAXONDriveStatus & *status*) [inline]

Parameters:

status[in] - ptr to structure representing hardware status bits

9.20.2.8 bool ACQ_Shutter::WriteShutterOpenClose (pfip::EPICSVariableLookupKey *varKey*, EPICSComm ∗const *epicsComm*)

Parameters:

setting[in] - the encoder value to use

epicsComm[in] - ptr to EPICS communication object

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

- pfip/include/ACQShutter.h
- pfip/src/ACQShutter.cpp

9.21 ACQ_SubSystem Class Reference

The ACQ_SubSystem class is used to manage the PFIP hardware, including the Probe Assembly, the Acquisition Camera, and the main Shutter. It uses the EPICS protocol to communicate with the hardware.

`#include <ACQSubSystem.h>`

Public Member Functions

- ACQ_SubSystem ()

  Constructor.

- virtual ~ACQ_SubSystem ()

  Destructor.

- void **StartStatusThread** ()
- void **StopStatusThread** ()
- void **StatusThreadEntryPoint** (void)
- PFIPConfig *const GetConfig ()

  Return ptr to PFIP configuration object.

- bool AttachEPICSContext ()

  Attach to the single EPICS context. This must be used in each thread that is accessing the Acq system hardware using EPICS.

- void **InitProbeMoveEventPayload** (const string &handlerName)
- void **SendProbeMoveEvent** (const string &handlerName)
- void **MarkProbeMoveEventComplete** (int retVal)
- bool **EnableGuideFilter** (const int probe)
- bool OpenShutter ()

  Open the acquisition camera shutter This is primarily for maintenance, as the shutter is usually controlled by the exposure process.

- bool CloseShutter ()

  Close the shutter Returns true if successful.

- bool HomeShutter ()

  Home the shutter Returns true if successful.

- pfip::FPAShutterState **GetShutterState** ()
- bool DeployACQCameraMirror ()

  Deploy the acquisition camera mirror Returns true if successful.

- bool RetractACQCameraMirror ()

  Retract the acquisition camera mirror Returns true if successful.

- bool DeployCWFSCameraMirror ()

  Deploy the Calibration Wave Front Sensor mirror Returns true if successful.

- pfip::MirrorSetting CWFSCameraMirrorState ()

Returns current state of the CWFS mirror.

- bool RetractCWFSCameraMirror ()

  Retract the Calibration Wave Front Sensor mirror Returns true if successful.

- bool EnableDisableProbes (const bool enableFlag)

  enable or disable all the configured probes

- bool ExposureSetup (const bool camraControlsFPShutter)

  Prepare for an exposure Check the state of the shutter -- cannot start an exposure while the shutter is busy with another exposure. Close the shutter, enable the acquisition camera filter. This method must be called before ann exposure can occur.

- int GetACQCameraFocus ()

  Return the setting of the acquisition camera focus.

- double GetACQCameraFocus_mm ()

  Return the setting of the acquisition camera focus in mm between 0.0 and 19.8.

- bool SetACQCameraFocus (const int value)
- bool SetACQCameraFocus_mm (const double value)
- pfip::MirrorSetting GetCWFSCameraState ()

  Return the current status of the CWFS camera. Returns true if the CWFS camera is deployed; false if the PV camera is deployed.

- pfip::MirrorSetting GetCWFSCameraDeployed ()

  Read the current setting for the CWFS/PV deployment from the hardware.

- bool SetCWFSCameraDeployed (const bool setting)

  Change the setting for the CWFS/PV deployment.

- void Interrupt ()

  Interrupt a long-running operation, such as enable/disable probes, move/home probes.

- bool **ProbeAssemblyIsActive** (void)
- int MoveProbes (const MoveProbeParms(&positions)[pfip::NUMBER_OF_PROBES])

  Move the probes to the x-y positions specified in the input structure If a probe is configured to be disabled, it will not be moved.

- bool ChangeProbeIGain (const pfip::ProbeID probeID, int iGain)

  Power on hardware components of the PFIP: Switch1, Switch2, SBC1, SBC2, Guiders, ACQCAM, PV/CWFS.

- bool **ChangeProbePGain** (const pfip::ProbeID probeID, int iGain)
- bool **TimeoutNextMove** (const double timeout)
- bool **PowerOnAll** ()
- bool PowerOffAll ()

  Power off hardware components of the PFIP: Switch1, Switch2, SBC1, SBC2, Guiders, ACQCAM, PV/CWFS.

- bool SetACQCameraPower (const pfip::PowerSetting val)

Power On/Off ACQ camera.

- bool SetACQCameraFilter (const int filterSetting)
  Change the acquisition camera filter setting.
- bool **SetACQCameraFilter** (const std::string &filter_name)
- bool SetSwitch1Power (const pfip::PowerSetting setting)
  manage the power setting for Ethernet Switch 1
- bool SetSwitch2Power (const pfip::PowerSetting setting)
  manage the power setting for Ethernet Switch 1
- bool SetPV_CWFSCameraPower (const pfip::PowerSetting setting)
  manage the power setting for Ethernet Switch 2
- bool SetTTCameraPower (const pfip::PowerSetting setting)
  manage the power setting for the TT camera source
- bool SetBIBCameraPower (const pfip::PowerSetting setting)
  manage power for the SBC1-3
- bool SetHPF_ACQCameraPower (const pfip::PowerSetting setting)
  manage power for the BIB Camera
- bool SetHR_SASCameraPower (const pfip::PowerSetting setting)
  manage power for the HPF_ACQ Camera
- bool SetSBC1Power (const pfip::PowerSetting setting)
  manage power for the HR_SAS Camera
- bool SetSBC2Power (const pfip::PowerSetting setting)
  manage power setting for the SBC4-5
- bool SetProbeAssemblyPower (const bool val)
  manage power setting for the SBC4-5
- bool SetProbePower (const pfip::ProbeID probe, const pfip::PowerSetting setting)
  manage power settings for the probes
- bool SetDither (const pfip::DitherPos setting)
  Change the dither setting.
- pfip::DitherPos **GetDither** ()
- bool SetENWAFilterPosition (const pfip::ENWAFilterPos pos)
  Change the Entry Window filter position.
- bool SetGuideFilter (const int probe, const pfip::GuiderFilterPos setting)
  Set the filter for guider 1 or 2 camera to setting.
- bool **SetGuideFilter** (const int probe, const std::string &name)

- void ShutDown ()

  Disable all probes and terminate EPICS connection.

- bool TakeExposure (const fptime &durationSec, const bool camraControlsFPShutter)

  take an exposure using the acquisition camera shutter and current filter setting

- pfip::ENWAFilterPos GetENWAFilterPosition ()

  Return the Entry Window position.

- pfip::EXWAFilterPos **GetEXWAFilterPosition** ()
- int **GetENWAEncoderPosition** ()
- Message * ReplyProbePosition (const pfip::ProbeID i)

  Return a message containing information about the specified probe and its position.

- Message * ReplyProbeInfo (const pfip::ProbeID i)

  Return a message containing information about the specified probe and filter.

- Message * ReplyACQStatus ()

  Return a message containing information about the acquisition structure status.

- void BroadcastStatusEvents ()

  Methods to broadcast status of component hardware.

- void **SendProbeStatusEvent** ()
- void **SendShutterStatusEvent** ()
- void **SendCameraStatusEvent** ()
- void **SendMiscStatusEvent** ()
- void **SendCWFSStatusEvent** ()
- void **SendENWAStatusEvent** ()
- double **GetShutterOpenDuration** (void)

9.21.1 Member Function Documentation

9.21.1.1 bool ACQ_SubSystem::EnableDisableProbes (const bool *enableFlag*)

Parameters:

enableFlag == true, enable, enableFlag == false, disable

Returns 1 if successful

Returns 0 on error condition

Returns -1 if interrupted

9.21.1.2 int ACQ_SubSystem::GetACQCameraFocus ()

Parameters:

Returns true if successful and exposure is ready to start

9.21.1.3 bool ACQ_SubSystem::SetACQCameraFilter (const int *filterSetting*)

Parameters:

filterSetting[in] - the new setting

9.21.1.4 bool ACQ_SubSystem::SetACQCameraFocus (const int *value*)

The input on the SetACQCameraFocus is an adjustment which is added to the current focus setting retrieved by GetACQACamFocus Returns true if successful.

9.21.1.5 bool ACQ_SubSystem::SetACQCameraFocus_mm (const double *value*)

An absolute setting for the acqcam focus between 0.0 and 19.8 mm.

9.21.1.6 bool ACQ_SubSystem::SetCWFSCameraDeployed (const bool *setting*)

Parameters:

setting - true for deploying CWFS, false for deploying pv

9.21.1.7 bool ACQ_SubSystem::SetDither (const pfip::DitherPos *setting*)

Parameters:

setting[in] - the new dither setting

9.21.1.8 bool ACQ_SubSystem::SetENWAFilterPosition (const pfip::ENWAFilterPos *pos*)

Parameters:

pos[in] - the new filter position

9.21.1.9 bool ACQ_SubSystem::TakeExposure (const fptime & *durationSec*, const bool *camraControlsFPShutter*)

Parameters:

durationSec[in] - number of seconds for exposure

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

- pfip/include/ACQSubSystem.h
- pfip/src/ACQSubSystem.cpp

9.22 ACQCamInfo Class Reference

Public Member Functions

- Message * **SendACQCamInfo** (CameraInfo *cam, ShutterInfo *shut, ENWAInfo *enwa, int ditherPosition)

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

- pfip/include/PFIPStatus.h
- pfip/src/EventInfo.cpp

9.23 AcqCamStatus Class Reference

Classes

- struct camera_status
- struct filter_status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- camera_status **m_AcqCamera**
- filter_status **m_AcqFilter**

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

- gui/include/pfipStatusData.h
- gui/src/pfipStatusData.cpp

9.24 Ui::acqCamStatusTab Class Reference

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

- gui/build/ui_acqCamStatusTab.h

9.25 acqCamStatusTabUi Class Reference

Signals

- void **ForceHeartbeat** (std::string)

Public Member Functions

- **acqCamStatusTabUi** (const char ∗name, QWidget ∗pLoadedWidget, guiConfig ∗pConfig)

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

- gui/include/acqCamStatusTabUi.h
- gui/build/moc_acqCamStatusTabUi.cpp
- gui/src/acqCamStatusTabUi.cpp

9.26 AcqCWFSStatus Class Reference

Classes

- struct cwfs_status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- cwfs_status **m_AcqCWFS**

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

- gui/include/pfipStatusData.h
- gui/src/pfipStatusData.cpp

9.27 AcqENWAStatus Class Reference

Classes

- struct enwa_status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- enwa_status **m_ENWA**

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

- gui/include/pfipStatusData.h
- gui/src/pfipStatusData.cpp

9.28 AcqMiscStatus Class Reference

Classes

- struct misc_status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- misc_status **m_AcqMisc**

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

- gui/include/pfipStatusData.h
- gui/src/pfipStatusData.cpp

9.29 AcqShutterStatus Class Reference

Classes

- struct shutter_status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- shutter_status **m_AcqShutter**

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

- gui/include/pfipStatusData.h
- gui/src/pfipStatusData.cpp

9.30 activate Class Reference

9.30.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **activate** (MetrologyLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.30.2 Member Function Documentation

9.30.2.1 void activate::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.31 add_correction Class Reference

9.31.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **add_correction** (MetrologyLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.31.2 Member Function Documentation

9.31.2.1 void add_correction::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.32 AdjustDither Class Reference

9.32.1 Detailed Description

Change the dither setting.

Required parameters:

pos<integer> - An integer between 1 and 3 (inclusive) that identifies the level of dithering.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **AdjustDither** (Receiver ∗rcv, ACQ_SubSystem ∗acq)
- virtual void HandlerAction (Message ∗pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message ∗pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.33 Alarm_Monitor Class Reference

9.33.1 Detailed Description

A class to manage reading alarms from a Modbus PLC

`#include <PFIPMonitor.h>`

Public Member Functions

- Alarm_Monitor (const string &monitorName, const string &enclName, int statusFrequency, int maxRegs)

  Create an object with a thread to read from the given ipaddress and port The monitorName is used to identify this item as the source for alarm alerts.

- void **StartEnvironmentMonitorThread** ()
- void StopEnvironmentMonitorThread ()

  Stop the thread.

- virtual void **checkEnvironment** ()=0
- MODBUSComm * GetCommPtr ()

  Return the modbus communication object.

- int **GetValue** (int regNum, unsigned short *const val)
- int **SetValue** (int regNum, unsigned short *const val)

Friends

- class VIRUS_Monitor
- class PFIP_Monitor

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

- pfip/include/PFIPMonitor.h
- pfip/src/PFIPMonitor.cpp

9.34 VirusMonitorStatus::alarms Struct Reference

Public Member Functions

- **BOOL_VAR** (PurgeAirTemp)
- **BOOL_VAR** (LowerEENCFluid)
- **BOOL_VAR** (MidEENCFluid)
- **BOOL_VAR** (VENCTemp)
- **BOOL_VAR** (EENCTemp)
- **BOOL_VAR** (VENCOverwrite)
- **BOOL_VAR** (VENCOther)
- **BOOL_VAR** (VENCHeatExchangerInletTemp)
- **BOOL_VAR** (VENCHeatExchangerOutletTemp)

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

- gui/include/pfipStatusData.h

9.35 PFIPMonitorStatus::alarms Struct Reference

Public Member Functions

- **BOOL_VAR** (UpperTemp)
- **BOOL_VAR** (LowerTemp)
- **BOOL_VAR** (UpperFluid)
- **BOOL_VAR** (LowerFluid)

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

- gui/include/pfipStatusData.h

9.36 Ui::alertsDialog Class Reference

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

- gui/build/ui_alertsDlg.h

9.37 alertsDialog Class Reference

Public Member Functions

- **alertsDialog** (QWidget ∗parent=0)

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

- gui/include/mainWindow.h

9.38 altaz_coordinate Class Reference

9.38.1 Detailed Description

Class representing azimuth and elevation coordinates.

`#include <tcs_coordinates.h>`

Public Member Functions

- **altaz_coordinate** (double alt, double az)

Public Attributes

- double **alt_**
- double az_

  altitude in degrees

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

- het_lib/common/include/tcs_coordinates.h

9.39 Amoeba< Func > Class Template Reference

template<typename Func> class Amoeba< Func >

Public Types

- enum { **N** = Func::NPars }
- enum { **M** = Func::NPars+1 }
- typedef Func::value_type **value_type**

Public Member Functions

- **Amoeba** (value_type ftol, int maxiter)
- void **optimize** (FMatrix< value_type, M, N > &p, FVector< value_type, M > &y, const Func &f)
- FVector< typename Func::value_type, Amoeba< Func >::M > **initialize_y** (FMatrix< value_type, M, N > &p, const Func &f) const

Protected Member Functions

- value_type amotry (FMatrix< value_type, M, N > &p, FVector< value_type, M > &y, FVector< value_type, N > &psum, const Func &f, const int ihi, const value_type fac)
- void **compute_psum** (FVector< value_type, N > &psum, const FMatrix< value_type, M, N > &p)

Protected Attributes

- value_type **ftol_**
- int **maxiter_**

9.39.1 Member Function Documentation

9.39.1.1 template<typename Func > Amoeba< Func >::value_type Amoeba< Func >::amotry (FMatrix< value_type, M, N > & *p*, FVector< value_type, M > & *y*, FVector< value_type, N > & *psum*, const Func & *f*, const int *ihi*, const value_type *fac*) [inline, protected]

Extrapolates by a factor fac through the face of the simplex across from the high point, tries it, and replaces the high point if the new point is better.

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

- het_lib/coord_trans/include/amoeba.h

9.40 Amp Class Reference

Class corresponding to a single CCD output amplifier.

`#include <camra_hardware.h>`

Public Member Functions

- **Amp** (int half, const CAMRAConfig &config, CCD *ccd)
- CCD & ccd (void)

  Returns reference to the CCD this Amp is associated with.

- const string & suffix (void) const

  Returns the filename suffix associated with this amplifier, "AAABC" -> <ifu id>=""><side><half> -> "010LU" -> IFU 10, left, upper.

- virtual void write (FitsOutWrapper &f, int rows, int cols)

  Method that writes the current data associated with this amplifier to the given FitsOut stream.

Protected Member Functions

- void **add_binned_header_fields** (FitsOutWrapper &f, int rows, int cols)

Protected Attributes

- CCD * **ccd_**
- uint16_t * **data_**
- string **suffix_**
- int **half_**
- string **half_str_**
- string **ccdsec_str_**
- string **detsec_str_**
- string **ampsec_str_**
- string **amplifie_str_**
- int **bias_**
- int **bias_target_**
- float **gain_**
- float **rdnoise_**
- string **prior_checksum_**

Friends

- class CAMRAHardware
- class Mux
- class Cntl
- class CCD

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

- camra/camra_hardware.h
- camra/camra_hardware.cpp

9.41 AnalysisRegion Struct Reference

Public Member Functions

- **AnalysisRegion** (int zero=0)
- **AnalysisRegion** (int xmin, int xmax, int ymin, int ymax)
- bool **operator==** (const AnalysisRegion &r)
- bool **operator!=** (const AnalysisRegion &r)

Public Attributes

- int **xmin**
- int **xmax**
- int **ymin**
- int **ymax**

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

- tcs/metrology.h
- tcs/metrology.cpp

9.42 apcCmd Class Reference

Public Member Functions

- **apcCmd** (const string &appName, apcCmdConfig *pConfig)
- virtual int **start** (void)
- apcCmdConfig * **getConfig** (void)

Protected Attributes

- Sender * **m_pApcServerCommandSender**
- apcCmdConfig * **m_pConfig**

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

- apc/include/apcCmd.h
- apc/src/apcCmd.cpp

9.43 apcCmdConfig Class Reference

Public Member Functions

- **apcCmdConfig** (bool isNamed=false)
- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

- virtual void print_summary (void)

  Called prior to displaying options.

9.43.1 Member Function Documentation

9.43.1.1 virtual void apcCmdConfig::print_summary (void) [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.43.1.2 void apcCmdConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- apc/include/apcCmdConfig.h
- apc/src/apcCmdConfig.cpp

9.44 apcConfig Class Reference

Public Member Functions

- **apcConfig** (bool isNamed=false)
- void **parsePDUConfigs** (void)
- void **parseGroupConfigs** (void)
- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

- virtual void print_summary (void)

  Called prior to displaying options.

Public Attributes

- double **m_statusThreadCycleTime_secs**
- int **m_numPDUs**
- string **m_guiTabNames** [10+1]
- string **m_PDU** [32+1]
- pduConfig * **m_pPDUConfig** [32]
- int **m_numGroups**
- string **m_Group** [10+1]
- GroupConfig * **m_pGroupConfig** [10]
- int **m_maxOutletsPerPDU**

9.44.1 Member Function Documentation

9.44.1.1 void apcConfig::print_summary (void) [virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.44.1.2 void apcConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- apc/include/apcConfig.h
- apc/src/apcConfig.cpp

9.45 APCData Class Reference

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- struct APCData::status **m_status**

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

- gui/include/apcStatusData.h
- gui/src/apcStatusData.cpp

9.46 APCException Class Reference

Public Member Functions

- **APCException** (tcsexcept::tcs_exception_t __apc, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.47 apcPduSubSystem Class Reference

Public Types

- enum **Cmd_t** {
  IDN, **USER_NAME**, **PASSWORD**, **STATUS**,
  ON, **OFF**, **VOLTAGE**, **POWER**,
  CURRENT }

Public Member Functions

- **apcPduSubSystem** (pduConfig ∗pConfig, double cycleTime)
- virtual void **registerReceiveMsgHandlers** (Receiver ∗pReceiver)
- virtual void **unregisterReceiveMsgHandlers** (Receiver ∗pReceiver)
- bool **getOutletStatus** (int outletNumber)
- **HANDLER_DECL** (Pdu, OnLine)
- **HANDLER_DECL** (Pdu, TurnOnOutlet, int outlet)
- **HANDLER_DECL** (Pdu, TurnOffOutlet, int outlet)
- **HANDLER_DECL** (Pdu, Status)
- **HANDLER_DECL** (Pdu, StatusOutlet, int outlet)

Protected Types

- enum **PduType_t** { **APC**, **APC7921B**, **TDK** }

Protected Member Functions

- virtual bool **initialize** (void)
- virtual double **statusWorker** (bool bForceEvent=false)
- virtual bool **checkReply** (int nCmd, unsigned int replyLength, unsigned char ∗replyString)
- virtual void **populateStatusPayload** (Payload ∗pPayload)
- virtual void **issueStatusEvent** (void)
- virtual void **setupCmds** (void)
- virtual bool **setOutlet** (int outlet, Cmd_t cmd, string action, Message ∗∗ppReply)

Protected Attributes

- pduConfig ∗ **m_pConfig**
- PduType_t **m_PDUType**
- map< Cmd_t, string > **m_cmd**
- bool **m_outletStatus** [16]
- int **m_voltage**
- int **m_powerInVoltAmps**
- int **m_powerInWatts**
- int **m_current**

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

- apc/include/apcPDUSubSystem.h
- apc/src/apcPDUSubSystem.cpp

9.48 apcServer Class Reference

Public Member Functions

- **apcServer** (const string &appName, apcConfig *pConfig)
- apcConfig * **getConfig** (void)
- virtual int **start** (void)
- virtual void **stop** (void)
- **SERVER_HANDLER_DECL** (GetConfiguration)
- **SERVER_HANDLER_DECL** (TurnOnOutletByName, string name)
- **SERVER_HANDLER_DECL** (TurnOffOutletByName, string name)
- **SERVER_HANDLER_DECL** (TurnOnGroup, string name)
- **SERVER_HANDLER_DECL** (TurnOffGroup, string name)
- **SERVER_HANDLER_DECL** (StatusOutletByName, string name)

Protected Member Functions

- void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- void **SetName** (string name)
- const char * **GetName** (void)
- void **SetDebug** (bool on)
- void **setStatusThreadCycleTime_ms** (int cycleTime_ms)
- int **getStatusThreadCycleTime_ms** (void)
- void **statusWorker** (void)
- string **findPdu** (string outlet, int &outletNumber)
- bool **issuePDUCommand** (const string &cmd, const int &outlet, string &Outlet_status)
- void **issueGroupStatusEvent** (void)
- bool **setOutletByName** (string name, string action, Message **ppReply)
- bool **setGroup** (string name, string action, Message **ppReply)

Protected Attributes

- apcConfig * **m_pConfig**
- Receiver * **m_pReceiver**
- Sender * **m_pPduSender**
- map< string, apcPduSubSystem * > **m_apcPduSubSystem**
- bool **m_Debug**
- string **m_Name**
- tthread::ThreadWorker * **m_StatusWorkerThread**
- int **m_StatusLoopCount**
- int **m_StatusThreadCycleTime_ms**

Friends

- class apcPduSubSystem
- class **tdkPduSubSystem**

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

- apc/include/apcServer.h
- apc/src/apcServer.cpp

9.49 Ui::apcTab Class Reference

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

- gui/build/ui_apcTab.h

9.50 apcTabUi Class Reference

Classes

- class tabData

Public Slots

- void **APCStartup** (void)
- void **UpdateStatus** (std::string, std::string, int, int, double, int, int)
- void **UpdateGroupStatus** (std::string, std::string)

Signals

- void **ForceHeartbeat** (std::string)
- void **TurnOnOutletByName** (std::string)
- void **TurnOffOutletByName** (std::string)
- void **TurnOnGroup** (std::string)
- void **TurnOffGroup** (std::string)
- void **CommandIssued** (QString, QString)
- void **CommandDone** (QString, bool)
- void **CommandError** (QString, QString)

Public Member Functions

- **apcTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Protected Types

- typedef QMap< string, QList< QLedIndicator * > > **PDULedMap_t**
- typedef QMap< string, QLedIndicator * > **GroupLedMap_t**
- typedef std::map< string, tabData * > **tabDataMap_t**

Protected Member Functions

- bool **constructTab** (void)
- void **deconstructTab** ()
- void **loadSettings** (void)
- void **saveSettings** (void)

Protected Attributes

- bool **m_bConstructTab**
- bool **m_bShowMenuBar**
- bool **m_bShowCurrent**
- bool **m_bShowGroups**
- Sender * **m_pApcCommandSender**

- QShortcut * **m_pToggleMenuBarShortcut**
- PDULedMap_t **m_PDULedMap**
- GroupLedMap_t **m_GroupLedMap**
- QList< QWidget * > **m_pTabPageWidgets**
- QList< QWidget * > **m_pOtherWidgets**
- tabDataMap_t **tabDataMap**

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

- gui/include/apcTabUi.h
- gui/build/moc_apcTabUi.cpp
- gui/src/apcTabUi.cpp

9.51 tcs_photometry::AperturePhotResults Struct Reference

9.51.1 Detailed Description

result of aperture photometry

`#include <object.h>`

Public Attributes

- float **f_**
- float **df_**
- float **m_**
- float **dm_**
- float **sky_**
- float **dsky_**
- float **skyMag_**

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

- het_lib/photometry/include/object.h

9.52 apply_offset Class Reference

9.52.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

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

- tcs/tcs_handler_doc.h

9.53 Archon Class Reference

9.53.1 Detailed Description

Class encapsulating assumed state based on past communication with the underlying archon controller.

`#include <archon.h>`

Public Member Functions

- void **init** (void)
- void **help** (void)

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

- ics/archon.h
- ics/archon.cpp

9.54 ArchonHandler Class Reference

Public Member Functions

- **ArchonHandler** (Archon &d, const MessageHandlerDescriptor &m)

Public Attributes

- Archon & **driver_**

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

- ics/archon.h

9.55 array_dbr_int_t Class Reference

Public Member Functions

- **array_dbr_int_t** (const std::string &name, const std::string &field_names)
- void **set** (const void *arg)
- virtual void **set** (const unsigned nfields, const void *arg)
- virtual void **apply_parms** (Payload &p) const

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

- epics_relay/array_types.h
- epics_relay/array_types.cpp

9.56 array_dbr_long_t Class Reference

Public Member Functions

- **array_dbr_long_t** (const std::string &name, const std::string &field_names)
- void **set** (const void *arg)
- virtual void **set** (const unsigned nfields, const void *arg)
- virtual void **apply_parms** (Payload &p) const

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

- epics_relay/array_types.h
- epics_relay/array_types.cpp

9.57 ArrayType Class Reference

Public Member Functions

- **ArrayType** (const std::string &name, const std::string &field_names)
- std::string **field_name** (const unsigned index) const
- unsigned **field_index** (const unsigned value) const
- const std::string & **prefix** (void) const
- unsigned **nfields** (void) const
- virtual void **set** (const void ∗arg)=0
- virtual void **set** (const unsigned nfields, const void ∗arg)=0

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

- epics_relay/array_types.h
- epics_relay/array_types.cpp

9.58 tcsdb::attr_descriptor Class Reference

Public Member Functions

- def **__init__**
- def **__call__**

Public Attributes

- **cursor_**
- **system_**
- **source_**
- **key_**
- **attr_**

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

- scripting/pytcs/build/lib.linux-x86_64-2.7/tcsdb.py

9.59 pytcs::tcsdb::attr_descriptor Class Reference

Public Member Functions

- def **__init__**
- def **__call__**

Public Attributes

- **cursor_**
- **system_**
- **source_**
- **key_**
- **attr_**

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

- scripting/pytcs/tcsdb.py

9.60 averager< Type > Class Template Reference

template<typename Type> class averager< Type >

Public Member Functions

- **averager** (unsigned window_length)
- void **set_window_length** (unsigned window_length)
- unsigned **get_window_length** (void)
- void **insert** (Type entry)
- Type **average** (void)
- void **clear** (void)

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

- het_lib/common/include/averager.h

9.61 AVT Struct Reference

Public Attributes

- char **addr** [100]
- tPvUint32 **heartbeatTimeout**
- int **numframes**
- int **packetSize**
- bool **bOpen**
- tPvUint32 **partNumber**
- unsigned int **addrval**
- tPvHandle **handle**
- struct {
  tPvUint32 **width**
  tPvUint32 **height**
  tPvUint32 **bitsperpixel**
  } **sensor**

- struct {
  tPvUint32 **startx**
  tPvUint32 **starty**
  tPvUint32 **width**
  tPvUint32 **height**
  tPvUint32 **binningx**
  tPvUint32 **binningy**
  } **image**

- bool **bBinningSupported**
- tPvFloat32 **exptime**
- tPvFloat32 **frametime**
- tPvUint32 **framesize**
- tPvUint32 **sbps**
- tPvFrame **frames** [100]
- tPvUint32 **tsfrequency**
- tPvCameraInfoEx **info**
- tPvIpSettings **ipsettings**
- struct {
  int **enabled**
  int **active**
  void * **r**
  void * **g**
  void * **b**
  } **bayer**

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

- pas/AVT_lib/include/libavt.h

9.62 BeginRangeProbes Class Reference

9.62.1 Detailed Description

Move all the probes and their arms to the beginning of their respective ranges. If one or more of the probes is diabled by configuration, that probe will not be affected. The PFIPInterrupt() handler can be used to cancel this operation.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **BeginRangeProbes** (Receiver ∗rcv, ACQ_SubSystem ∗acq)
- virtual void HandlerAction (Message ∗pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message ∗pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.63 bestDistance Struct Reference

Public Types

- enum { **NPars** = 2 }
- typedef double **value_type**

Public Member Functions

- **bestDistance** (pfip_coord &pb, double xi_x, double xi_y)
- template<int M>
  double **operator()** (const ltl::FVector< double, 2, M > &angles) const

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

- het_lib/coord_trans/src/pfip.cpp

9.64 ExposureInfo::BINNING Struct Reference

Public Attributes

- int **width**
- int **height**

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

- pas/include/Camera.h

9.65 Bitfield Class Reference

Public Member Functions

- **Bitfield** (const std::string &name, const std::string &field_names)
- void **set** (const void *arg)
- virtual void **set** (const unsigned nfields, const void *arg)
- virtual void **apply_parms** (Payload &p) const

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

- epics_relay/array_types.h
- epics_relay/array_types.cpp

9.66 BuildPupilThreadArgs Struct Reference

Public Attributes

- void * **myclass**
- ulong **thrstart**
- ulong **thrstride**

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

- het_lib/wfs_lib/include/wfslib.h

9.67 BusyHandler Class Reference

Public Member Functions

- virtual void process (Message *message)

9.67.1 Member Function Documentation

9.67.1.1 virtual void BusyHandler::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- het_lib/messaging/include/builtin_handlers.h

9.68 Camera Class Reference

Base Camera class.

`#include <Camera.h>`

Public Member Functions

- **Camera** (CameraConfig *pConfig, pasImagePipelineSubSystem *pSubSystem)
- Camera_t **GetType** (void)
- std::string **GetTypeName** (void)
- virtual bool **SetPixelFormat** (PixelFormat_t type)
- virtual PixelFormat_t **GetExposureType** (void)
- virtual bool **SetImageType** (Image_t type)
- virtual Image_t **GetImageType** (void)
- virtual bool **SetExposureTime** (float time)
- virtual double **GetExposureTime** (void)
- virtual bool **SetOverScan** (int overScan)
- virtual int **GetOverScan** (void)
- virtual bool **SetNumFlushes** (int numFlushes)
- virtual bool **SetObjectAndMagnitudes** (Message *pMessage, string object)
- virtual void **ClearObjectAndMagnitudes** (void)
- virtual bool **SetBinning** (int width, int height)
- virtual bool **GetBinning** (int *width, int *height)
- virtual bool **SetRegionOfInterest** (int x, int y, int width, int height)
- virtual bool **GetRegionOfInterest** (int *x, int *y, int *width, int *height)
- virtual bool **SetReadoutSpeed** (ReadoutSpeed_t)
- virtual bool **GetGain** (int *)
- virtual bool **SetTargetTemperature** (double)
- virtual bool **GetTemperatures** (double *, double *, double *)
- virtual bool **StatusChanged** (void)
- virtual void **PopulateStatusPayload** (Payload &payload)
- virtual void **TakeControl** ()
- virtual void **RelinquishControl** (void)

Protected Member Functions

- virtual bool **attemptReconnect** (void)
- virtual bool **initialize** (void)
- virtual bool **uninitialize** (void)
- virtual bool **closeCamera** (void)
- virtual void **socketStatusWorker** (void)
- virtual bool **isOpen** (void)=0
- virtual bool **setBinning** (void)
- virtual bool **setRegionOfInterest** (void)
- virtual bool **setExposureTime** (void)
- virtual bool **setOverScan** (void)
- virtual bool **setPixelFormat** (void)
- virtual bool **setImageType** (void)
- virtual bool **exposeFrame** (void)=0

- virtual void **stopExposure** (void)
- virtual bool **startTakingImages** (void)
- virtual bool **stopTakingImages** (void)
- virtual int **getExposureTimeLeft** (void)=0
- bool **SetFilter** (void)
- bool **ExposeFrame** (void)
- ExposureInfo * **GetAndLockExposureInfo** (bool bForce=0)
- void **UnlockExposureInfo** (void)
- double **GetExposureOverhead** (void)
- double **GetExposureTimeWithFilterFactor** (bool bLockExposureMutex)
- int **GetNumFlushes** (void)
- void **updateTemperaturesAndGain** (void)
- bool **notImplementedError** (const char *func)
- bool **notImplementedInfo** (const char *func)

Protected Attributes

- std::string **m_socketAddress**
- tthread::mutex **m_exposureInfoAccessMutex**
- ExposureInfo **m_exposureInfo**
- ExposureInfo **m_lastExposureInfo**
- unsigned int **m_exposureTimeLeft**
- Camera_t **m_type**
- std::string **m_typeName**
- bool **m_bOpen**
- bool **m_bExposureInProgress**
- bool **m_bCaptureInProgress**
- CameraConfig * **m_pConfig**
- pasImagePipelineSubSystem * **m_pSubSystem**
- TCSSocket * **m_pSocket**
- tthread::ThreadWorker * **m_socketStatusWorkerThread**
- bool **m_bEmptySocketBeforeNextCmd**
- tthread::mutex **m_cameraAccessMutex**

Friends

- class pasImagePipelineSubSystem

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

- pas/include/Camera.h
- pas/src/Camera.cpp

9.69 AcqCamStatus::camera_status Struct Reference

Public Member Functions

- **DOUBLE_VAR** (focus, 6, 2)
- QString **mirror** (void)

Public Attributes

- flags **m_flags**
- int **m_mirror**

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

- gui/include/pfipStatusData.h

9.70 CameraAVT Class Reference

Public Member Functions

- **CameraAVT** (CameraConfig *pConfig, pasImagePipelineSubSystem *pSubSystem)
- virtual bool **SetReadoutSpeed** (ReadoutSpeed_t speed)
- virtual bool **GetGain** (int *gain)
- virtual bool **GetTemperatures** (double *targetTempC, double *tempC, double *elecTempC)
- void **FrameCallback** (tPvFrame *pFrame)

Protected Member Functions

- virtual bool **initialize** (void)
- virtual bool **uninitialize** (void)
- virtual bool **closeCamera** (void)
- virtual bool **isOpen** (void)
- virtual void **socketStatusWorker** (void)
- virtual bool **setBinning** (void)
- virtual bool **setRegionOfInterest** (void)
- virtual bool **setExposureTime** (void)
- virtual bool **setPixelFormat** (void)
- virtual bool **exposeFrame** (void)
- virtual int **getExposureTimeLeft** (void)
- virtual void **stopExposure** (void)
- void **cameraError** (const char *function, tPvErr rc, bool bIgnoreError=0)
- bool **cameraNotOpen** (const char *function)

Protected Attributes

- AVT **m_AVT**
- unsigned **m_numFrames**

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

- pas/include/CameraAVT.h
- pas/src/CameraAVT.cpp

9.71 CameraConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- string **Guider1Camera_address**
- string **Guider2Camera_address**
- string **WFS1Camera_address**
- string **WFS2Camera_address**
- string **CWFSCamera_address**
- string **DWFS1Camera_address**
- string **DWFS2Camera_address**
- string **DWFS3Camera_address**
- string **DWFS4Camera_address**
- string **DWFS5Camera_address**
- string **DWFS6Camera_address**
- string **DWFS7Camera_address**
- string **DWFS8Camera_address**
- string **DWFS9Camera_address**
- string **DWFS10Camera_address**
- string **DWFS11Camera_address**
- string **DWFS12Camera_address**
- string **DWFS13Camera_address**
- string **ACQCamera_address**
- string **PVCamera_address**
- string **DMVCamera_address**
- string **BIBCamera_address**
- string **HPF_ACQCamera_address**
- string **VATCamera_address**
- string **CATCamera_address**
- string **TTCamera_address**
- string m_imageSourcePath

  Simulated image source path.

- bool **m_bCycleImageFiles**
- string m_manufac

  Camera manufacturer ("FLI", "AVT", "PGR", "SIM").

- string **m_model**
- double m_heartbeatTimeout

  Camera activity timeout (AVT only).

- int m_exposureCount

  Starting exposure count for simulated file name.

- int m_format [2+1]

  CCD configuration (widht x cols).

- int **m_overScan**
- int m_wcsRefPixel [2+1]

  WCS reference pixel (x, y).

- double m_pixelSize

  Bits per pixel.

- int m_packetSize

  Maximum TCP packet size.

- int m_streamBPS

  Stream bytes per second (height ∗ width ∗ frameRate ∗ bytesPerPixel).

- int **m_darkCurrentCompensation**
- int m_binWidth

  Binning pixels per column.

- int m_binHeight

  Binning pixels per row.

- int **m_numFilters**
- string m_filter [5+1]

  --filters option - list of filter(s) in wheel order

- double m_filterExposureFactor [5+1]

  --filter-exposure-factors - list of filter exposure factor(s)

- double m_filterZeroPoint [5+1]

  --filter-zero-points - list of filter zero point(s)

- int **m_filterSequenceLength**
- int m_filterSequence [5+1]

  --filter-sequence - list of filter(s) to use in sequence

- double m_expTime

  Exposure time in seconds.

- int **m_numPixelFormats**
- string **m_pixelFormats** [pas::PIXEL_FORMAT_MAX_COUNT+1]
- string **m_pixelFormat**
- pas::PixelFormat_t **m_pixelFormatType**
- double m_expOverhead

  Exposure overhead per frame in seconds.

- double m_targetTemp

  Target temperature (C).

- int m_flushes

  Number of frames to discard for each exposure.

9.71.1 Member Function Documentation

9.71.1.1 void CameraConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

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

- pas/include/pasConfig.h
- pas/src/pasConfig.cpp

9.72 Ui::cameraControlTab Class Reference

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

- gui/build/ui_cameraControlTab.h

9.73 cameraControlTabUi Class Reference

Signals

- void **ForcePipelineStatusEvent** (std::string)
- void **SetTargetTemp** (std::string, double)
- void **SetExposureTime** (std::string, double)
- void **SetIntervalSpeed** (std::string, double)
- void **SetCoAddImages** (std::string, int)
- void **SetBinning** (std::string, int, int)
- void **SetNumFlushes** (std::string, int)
- void **SetCleanImages** (std::string, bool)
- void **SetProcessImages** (std::string, bool)
- void **SetStoreImages** (std::string, bool)
- void **RelayViewImageExtensionChanged** (std::string, int)
- void **UpdateSetup** (bool)

Public Member Functions

- **cameraControlTabUi** (const char ∗name, QWidget ∗pLoadedWidget, guiConfig ∗pConfig, char ∗pipelineName)

Protected Attributes

- string **m_pipelineName**

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

- gui/include/cameraControlTabUi.h
- gui/build/moc_cameraControlTabUi.cpp
- gui/src/cameraControlTabUi.cpp

9.74 CameraFLI::CameraData Struct Reference

Public Attributes

- int **hm**
- int **vm**
- int **nx**
- int **ny**
- int **nb**
- float **ccdTemp**
- float **coolerPower**
- int **warmedUp**
- float **baseTemp**

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

- pas/include/CameraFLI.h

9.75 CameraFLI Class Reference

Classes

- struct CameraData

Public Member Functions

- **CameraFLI** (CameraConfig *pConfig, pasImagePipelineSubSystem *pSubSystem)
- virtual bool **SetReadoutSpeed** (ReadoutSpeed_t speed)
- virtual bool **GetGain** (int *gain)
- virtual bool **SetTargetTemperature** (double tempC)
- virtual bool **GetTemperatures** (double *targetTempC, double *tempC, double *elecTempC)

Protected Member Functions

- virtual bool **initialize** (void)
- virtual bool **uninitialize** (void)
- virtual bool **closeCamera** (void)
- virtual void **socketStatusWorker** (void)
- virtual bool **isOpen** (void)
- virtual bool **exposeFrame** (void)
- virtual bool **setExposureTime** (void)
- virtual int **getExposureTimeLeft** (void)
- virtual bool **SetRegionOfInterest** (int x, int y, int width, int height)
- virtual bool **setBinning** (void)
- virtual bool **setOverScan** (void)
- bool **sendCmdCheckReply** (char *cmdStr, unsigned int timeout_sec=5)
- bool **getReply** (char *cmdString, unsigned char *pBuf, int nBufLen, unsigned int timeout_sec=5)
- bool **getImage** (void)
- bool **setReadoutSpeed** (ReadoutSpeed_t speed)

Protected Attributes

- char **m_response** [100+1]
- CameraData **m_cameraData**
- float **m_targetTemp**
- ReadoutSpeed_t **m_readoutSpeed**
- char **m_serialNum** [81]
- char **m_model** [81]

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

- pas/include/CameraFLI.h
- pas/src/CameraFLI.cpp

9.76 cameraImageMarker Class Reference

Public Types

- enum **markerShape_t** { **circle**, **rectangle**, **crosshair**, **x** }

Public Member Functions

- QPointF **center** (void)

Public Attributes

- QString **m_name**
- QString **m_text**
- markerShape_t **m_shape**
- QPointF **m_externalTopLeftOrCenter**
- qreal **m_widthOrRadius**
- qreal **m_height**
- QColor **m_color**
- QAbstractGraphicsShapeItem ∗ **m_pItem**

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

- gui/include/cameraImageScene.h
- gui/src/cameraImageScene.cpp

9.77 cameraImageScene Class Reference

Public Slots

- void **RaDecChanged** (double, double, double, double, double, double, double, double, double)

Signals

- void **OffsetTelescope** (double, double, bool, bool)
- void **OffsetProbe** (std::string, double, double)
- void **SetFiducial** (std::string, double, double)

Public Member Functions

- **cameraImageScene** (cameraImageTabUi *pTab, QGraphicsView *pView)
- QGraphicsPixmapItem * **replaceImagePixmap** (QPixmap *pPixmap)
- QGraphicsPixmapItem * **getImagePixmap** (void)
- cameraImageTabUi * **getParentTab** ()
- void **removeImagePixmap** (void)
- double **getScaleFactor** (void)
- void **enableScaling** (bool enabled)
- void **enableDrawing** (bool enabled)
- void **enableContrastAdjusting** (bool enabled)
- void **enableMonitorPosition** (bool enabled)
- void **enableMouseScrolling** (bool enabled)
- bool **getGuideBox** (imageRectI &imageRect)
- bool **getManualCentroidBox** (imageRectI &imageRect)
- bool **getManualCentroid** (QPointF &center)
- bool **getInstrumentCentroid** (QPointF &center)
- bool **getManualFiducial** (imageRectI &imageRect)
- bool **getInstrumentFiducial** (imageRectI &imageRect)
- void **setPlateScale** (double plateScale)
- void **setOrientation** (double orientation)
- void **setBinning** (int width, int height)
- void **showDialogs** (bool show=true)
- void **replaceCursor** (const QPointF &center)
- void **removeCursor** (bool bUnsetCursor=true)
- void **replaceInstrumentCentroid** (const QPointF &center)
- void **replaceManualCentroid** (const QPointF &center)
- void **replaceManualFiducial** (const QPointF &center)
- void **replaceInstrumentFiducial** (const QPointF &center)
- void **replaceGuideBox** (const QPointF &endingPoint, const QPointF &centerPoint)
- void **removeGuideBox** (void)
- void **removeManualCentroidBox** (void)
- void **removeManualCentroid** (void)
- void **removeManualFiducial** (void)
- void **removeMeasuringTriangle** (void)
- void **addMarker** (cameraImageMarker *pMarker)
- void **removeMarkers** (void)

- void **removeFromToMarkers** (void)
- void **addRose** (double orientation, double parallacticAngle, double rhoOffset, QPointF centerPoint, double xLength, double yLength)
- void **addGraphicOverlays** (void)

Protected Member Functions

- virtual void **mousePressEvent** (QGraphicsSceneMouseEvent ∗event)
- virtual void **mouseMoveEvent** (QGraphicsSceneMouseEvent ∗event)
- virtual void **mouseReleaseEvent** (QGraphicsSceneMouseEvent ∗pEvent)
- virtual void **wheelEvent** (QGraphicsSceneWheelEvent ∗event)
- double **get_current_orientation** (void)
- void **offsetTrajectory** (const QPointF &delta, MoveWhich_t moveWhich)
- cameraImageMarker ∗ **findMarker** (QPointF cursorPos)
- void **adjustImageContrast** (const QPointF &scenePos)
- void **showGuideBoxDialog** (void)
- void **showFiducialDialog** (void)
- QGraphicsRectItem ∗ **drawBox** (const QColor &color, const QPointF &endingPoint, const QPointF &centerPoint=QPointF(-1,-1))
- void **drawMarker** (QList< QGraphicsItem ∗ > &itemLis, cameraImageMarker ∗pMarker)
- void **drawMeasuringTriangle** (QList< QGraphicsItem ∗ > &itemList, const QPointF &endingPoint)
- void **drawRose** (QList< QGraphicsItem ∗ > &itemList, double orientation, double parallacticAngle, double rhoOffset, QPointF centerPoint, double xLength, double yLength)
- void **deleteItemsInList** (QList< QGraphicsItem ∗ > &itemList)
- void **drawGridOverlay** (void)
- void **drawM1GridOverlay** (void)
- void **drawPupilOverlay** (void)
- void **drawObscurationOverlay** (void)
- void **drawVectorsOverlay** (void)

Protected Attributes

- bool **m_bScalingEnabled**
- bool **m_bDrawingEnabled**
- bool **m_bContrastAdjustingEnabled**
- bool **m_bMonitorPositionEnabled**
- bool **m_bMouseScrollingEnabled**
- bool **m_bDrawingGuideBox**
- bool **m_bDrawingCentroidBox**
- bool **m_bDrawingManualFiducial**
- bool **m_bDrawingManualFiducialAtMarker**
- bool **m_bManualCentroiding**
- bool **m_bMoveToMarker**
- bool **m_bMoveToManualFiducial**
- bool **m_bMoveToInstrumentFiducial**
- MoveWhich_t **m_moveWhich**
- bool **m_bMeasuring**
- bool **m_bContrastAdjusting**

- bool **m_bMouseScrolling**
- bool **m_bCentering**
- Photometry * **m_pPhotometry**
- QPointF **m_startingPoint**
- cameraImageTabUi * **m_pTab**
- QGraphicsView * **m_pView**
- QGraphicsPixmapItem * **m_pPixmapItem**
- double **m_scaleFactor**
- QGraphicsPathItem * **m_pCursor**
- QGraphicsRectItem * **m_pGuideBox**
- QGraphicsRectItem * **m_pManualCentroidBox**
- QGraphicsPathItem * **m_pInstrumentCentroid**
- QGraphicsPathItem * **m_pManualCentroid**
- QPointF **m_manualCentroidCenter**
- QPointF **m_instrumentCentroidCenter**
- QGraphicsPathItem * **m_pManualFiducial**
- QGraphicsPathItem * **m_pInstrumentFiducial**
- guideBoxDialog * **m_pGuideBoxDialog**
- fiducialDialog * **m_pFiducialDialog**
- QList< QGraphicsItem * > **m_measuringTriangleList**
- QList< QGraphicsItem * > **m_markerList**
- QList< QGraphicsItem * > **m_fromToMarkerList**
- QList< QGraphicsItem * > **m_roseList**
- QList< QGraphicsItem * > **m_shGridOverlayList**
- QList< QGraphicsItem * > **m_m1OverlayList**
- QList< QGraphicsItem * > **m_pupilOverlayList**
- QList< QGraphicsItem * > **m_obscOverlayList**
- QList< QGraphicsItem * > **m_shVectorsDataList**
- double **m_dec**
- double **m_plateScale**
- double **m_orientation**
- int **m_binWidth**
- int **m_binHeight**

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

- gui/include/cameraImageScene.h
- gui/build/moc_cameraImageScene.cpp
- gui/src/cameraImageScene.cpp

9.78 Ui::cameraImageTab Class Reference

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

- gui/build/ui_cameraImageTab.h

9.79 cameraImageTabUi Class Reference

Public Types

- typedef QMap< QString, cameraImageMarker * > **markerMapEntry_t**
- typedef QMap< QString, markerMapEntry_t * > **markerMap_t**

Signals

- void **UpdatePositionIndicator** (int, int)
- void **SetAnalysisRegion** (std::string, imageRectI)
- void **StartGuiding** (std::string)
- void **StopGuiding** (std::string)
- void **SetFiducial** (std::string, double, double)
- void **ClearFiducial** (std::string)
- void **TakeSingleImage** (std::string)

Public Member Functions

- **cameraImageTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig, char *pipelineName)
- void **updatePositionInfo** (int x, int y)
- int **getImageWidth** (void)
- int **getImageHeight** (void)
- string **getPipelineName** (void)
- bool **getConfirmOffsetDialog** (void)

Public Attributes

- cameraImageScene * **m_pCameraImageScene**
- cameraImageScene * **m_pFullCameraImageScene**
- cameraImageScene * **m_pWhiteImageScene**
- double **m_imageGainFactor**
- double **m_imageBiasFactor**
- Sender * **m_pImagesAsyncCommandSender**
- Sender * **m_pPASCmdSender**
- msgid_t **m_ImageMsgID**

Protected Member Functions

- virtual void **showDialogs** (bool show=true)
- void **loadSettings** (void)
- void **loadMarkerSettings** (void)
- void **loadLocalPhotometrySettings** (void)
- void **saveSettings** (void)
- void **saveLocalPhotometrySettings** (void)
- bool **eventFilter** (QObject *pObject, QEvent *pEvent)
- **DEFINE_THREAD** (cameraImageTabUi, ReceiveImagesAsyncReplies)

Protected Attributes

- int **m_subApertureOverlaySize**
- int **m_pupilOverlaySize**
- int **m_obscOverlaySize**
- int **m_shVectorsDataSize**
- int **m_m1OverlaySize**
- SHGrid **m_subApertureOverlay**
- PupilData **m_pupilOverlay**
- ObscGeom **m_obscOverlay**
- SHTrackerData **m_shVectorsData**
- SHGrid **m_m1Overlay**
- bool **m_bViewSubApertureGraphicOverlay**
- bool **m_bViewM1GraphicOverlay**
- bool **m_bViewPupilGraphicOverlay**
- bool **m_bViewObscurationGraphicOverlay**
- bool **m_bViewVectorsGraphicOverlay**
- markerMap_t **m_markerMap**
- string **m_pipelineName**
- bool **m_bConfirmOffsetDialog**
- pas::Camera_t **m_cameraType**
- pas::ProcessingMode_t **m_processingMode**
- photometryOptionsDialog * **m_pPhotometryOptionsDialog**
- bool **m_bFirstImage**
- bool **m_bImageSizeChanged**
- QImage **m_origImage**
- QImage **m_origAdjImage**
- QImage **m_currentImage**
- double **m_contrastScale**
- double **m_minPixelValue**
- tcs_gui::imageTypes **m_viewImageExtension**
- double **m_ra**
- double **m_dec**
- double **m_plateScale**
- double **m_orientation**
- double **m_parallacticAngle**
- tcs_photometry::ConfigureParams **m_photometryOptions**

Friends

- class cameraImageTabUiThreadWorker
- class cameraImageScene
- class confirmOffsetDialog

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

- gui/include/cameraImageTabUi.h
- gui/build/moc_cameraImageTabUi.cpp
- gui/src/cameraImageTabUi.cpp

9.80 cameraImageTabUiThreadWorker Class Reference

Signals

- void **NewImage** (int, const QImage &, const QImage &, double, double, double, double, double, double, double, bool, double, bool, double)
- void **CommandIssued** (QString, QString)
- void **CommandDone** (QString, bool)
- void **CommandError** (QString, QString)
- void **RelaySetViewImageExtension** (std::string, int)

Public Member Functions

- **cameraImageTabUiThreadWorker** (const char ∗name, cameraImageTabUi ∗pParent)
- void **ReceiveImagesAsyncReplies** (void)

Protected Member Functions

- void **loadGraphicOverlays** (unsigned char ∗pBuffer)
- void **extractCurrentControlValues** (ltl::FitsIn &fitsIn)

Protected Attributes

- cameraImageTabUi ∗ **m_pParent**
- CBUF< unsigned, 10, double > **m_minHistory**
- CBUF< unsigned, 10, double > **m_maxHistory**

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

- gui/include/cameraImageTabUi.h
- gui/build/moc_cameraImageTabUi.cpp
- gui/src/cameraImageTabUi.cpp

9.81 CameraInfo Class Reference

Public Member Functions

- void **SetCameraStatus** (MAXONDriveStatus &s)
- void **SetCameraFocus** (int foc)
- void **SetCameraFocus_mm** (double foc_mm)
- void **SetCameraFilterPos** (int Pos)
- void **SetCameraMirror** (int mirror)
- void **SetCameraState** (int state)
- void **SetCameraPower** (bool power)
- void **SendCameraInfo** ()
- void **SetCameraFilterName** (std::string name)
- void **SetCameraFilterConfig** (std::string configs)
- int **GetCameraFilterPos** ()
- int **GetCameraMirrorPos** ()
- int **GetCameraFocus** ()
- double **GetCameraFocus_mm** ()
- bool **GetCameraPower** ()

Public Attributes

- char **name** [256]
- uint **pixformat**
- uint **binsize**
- float **exposure**
- uint **roix**
- uint **roiy**
- uint **roiwidth**
- uint **roiheight**

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

- het_lib/wfs_lib/include/wfslib.h
- pfip/include/PFIPStatus.h
- pfip/src/EventInfo.cpp

9.82 Ui::cameraMetrologyControlTab Class Reference

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

- gui/build/ui_cameraMetrologyControlTab.h

9.83 cameraMetrologyControlTabUi Class Reference

Signals

- void **AddCorrections** (double, double, double, double, double, double)
- void **SetProcessingType** (std::string, std::string)
- void **CommandIssued** (QString, QString)
- void **CommandDone** (QString, bool)
- void **CommandError** (QString, QString)
- void **StartPipeline** (std::string)
- void **StopPipeline** (std::string)
- void **RelayCameraImageCommand** (std::string, std::string)
- void **StartMetrologyLoop** (std::string)
- void **StopMetrologyLoop** (std::string)
- void **ActivateMetrologyLoop** (std::string)
- void **DeactivateMetrologyLoop** (std::string)
- void **CenterProbe** (std::string)
- void **ForceLoopStateEvent** (std::string)
- void **ForcePipelineStatusEvent** (std::string)
- void **AddCorrections** (std::string, double, double, double, double, double, double)
- void **OffsetProbe** (std::string, double, double)
- void **ResetMetrologyLoop** (std::string)

Public Member Functions

- **cameraMetrologyControlTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig, char *pipelineName)

Protected Attributes

- string **m_pipelineName**
- Sender * **m_pPASCommandSender**
- photometryOptionsDialog * **m_pPhotometryOptionsDialog**
- tcs_photometry::ConfigureParams **m_photometryOptions**

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

- gui/include/cameraMetrologyControlTabUi.h
- gui/build/moc_cameraMetrologyControlTabUi.cpp
- gui/src/cameraMetrologyControlTabUi.cpp

9.84 Ui::cameraMetrologyDataTab Class Reference

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

- gui/build/ui_cameraMetrologyDataTab.h

9.85 cameraMetrologyDataTabUi Class Reference

Public Member Functions

- **cameraMetrologyDataTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig, char *pipelineName)

Protected Member Functions

- void **showWidgetsForThisPipelineType** (pas::ProcessingMode_t processingMode)

Protected Attributes

- string **m_pipelineName**
- pas::ProcessingMode_t **m_processingMode**

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

- gui/include/cameraMetrologyDataTabUi.h
- gui/src/cameraMetrologyDataTabUi.cpp

9.86 CameraPGR Class Reference

Public Member Functions

- **CameraPGR** (CameraConfig *pConfig, pasImagePipelineSubSystem *pSubSystem)
- virtual bool **SetExposureTime** (float time)
- virtual bool **SetImageType** (Image_t type)
- virtual bool **SetReadoutSpeed** (ReadoutSpeed_t speed)
- virtual bool **GetGain** (int *gain)
- virtual bool **GetTemperatures** (double *targetTempC, double *tempC, double *elecTempC)
- void **FrameCallback** (FlyCapture2::Image *pImage)

Protected Member Functions

- virtual bool **initialize** (void)
- virtual bool **uninitialize** (void)
- virtual bool **closeCamera** (void)
- virtual bool **isOpen** (void)
- virtual void **socketStatusWorker** (void)
- virtual bool **startTakingImages** (void)
- virtual bool **stopTakingImages** (void)
- virtual int **getExposureTimeLeft** (void)
- virtual void **stopExposure** (void)
- virtual bool **setBinning** (void)
- virtual bool **setRegionOfInterest** (void)
- virtual bool **setExposureTime** (void)
- virtual bool **setPixelFormat** (void)
- virtual bool **exposeFrame** (void)
- void **cameraError** (const char *function, FlyCapture2::Error &9210)
- bool **cameraNotOpen** (const char *function)
- bool **turnOffFrameRate** (void)
- bool **turnOffAutoExposure** (void)
- bool **turnOffWhiteBalance** (void)
- bool **turnOffBrightness** (void)
- bool **turnOffGamma** (void)
- bool **turnOffSharpness** (void)
- bool **setGain** (float gain)
- bool **setPacketSize** (int size)

Protected Attributes

- FlyCapture2::BusManager **m_busManager**
- FlyCapture2::PGRGuid **m_guid**
- FlyCapture2::GigECamera **m_camera**
- FlyCapture2::CameraInfo **m_cameraInfo**
- FlyCapture2::CameraStats **m_cameraStats**
- FlyCapture2::GigEImageSettings **m_imageSettings**
- FlyCapture2::GigEImageSettingsInfo **m_imageSettingsInfo**
- bool **m_bConnected**

- unsigned **m_numFrames**

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

- pas/include/CameraPGR.h
- pas/src/CameraPGR.cpp

9.87 CameraSim Class Reference

Public Member Functions

- **CameraSim** (CameraConfig *pConfig, pasImagePipelineSubSystem *pSubSystem)
- virtual bool **GetGain** (int *gain)
- virtual bool **SetTargetTemperature** (double tempC)
- virtual bool **GetTemperatures** (double *targetTempC, double *tempC, double *elecTempC)

Protected Member Functions

- virtual bool **initialize** (void)
- virtual bool **isOpen** (void)
- virtual bool **setBinning** (void)
- virtual bool **setRegionOfInterest** (void)
- virtual bool **setExposureTime** (void)
- virtual bool **setOverScan** (void)
- virtual bool **setPixelFormat** (void)
- virtual bool **setImageType** (void)
- virtual bool **exposeFrame** (void)
- virtual int **getExposureTimeLeft** (void)

Protected Attributes

- bool **m_bOpen**
- int **m_exposureCount**
- long **m_nAxes** [3]
- int **m_nAxis**
- double **m_targetTemperature**
- double **m_cameraTemperature**
- bool **m_imageFound**

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

- pas/include/CameraSim.h
- pas/src/CameraSim.cpp

9.88 CAMRAConfig Class Reference

Public Member Functions

- **CAMRAConfig** (bool is_named)
- void print_summary (void)

  Called prior to displaying options.

- void set_options (OptionParser &o)

  Adds options to default set.

- void **parse_spectrograph_configs** (string root)
- void **parse_controller_configs** (string root)

Public Attributes

- string **instrument**
- string **output_root**
- string **storage_root**
- string **output_prefix**
- string **shutter_device**
- bool **lab_mode**
- float **status_period**
- string **data_handler_route**
- bool **shutter_pfip**
- bool **enable_leds**
- float **readout_delay**
- bool **skip_onewire_read**

Global setpoints

Global setpoints for the CCD subsystem. These are overriden on an individual controller and/or CCD basis in subsequent configuration layers.

- string **dsp_timing_file**
- float **temp_setpoint**

Per-Controller setpoints

Optional configuration information for each spectrograph that overrides that found in the global setpoints.

- string **spectrograph_config_files** [96+1]
- std::map< int, SpectrographConfig * > **spectrograph_configs**
- string **controller_config_files** [96+1]
- std::map< int, ControllerConfig * > **controller_configs**

9.88.1 Member Function Documentation

9.88.1.1 void CAMRAConfig::print_summary (void) [virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.88.1.2 void CAMRAConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- camra/camra_config.h
- camra/camra_config.cpp

9.89 CAMRADriver Class Reference

Encapsulates ARC CArcSys class for remote access.

9.89.1 Detailed Description

CCD "driver" class that encapsulate all interfaces to the ARC hardware that are exposed to a remote client. The class initializes the sub-system receiver and provides a `start()` method for opening up the sub-system for remote commands. The handle on the ARC hardware is kept within the scope of the class and all interaction with the ARC hardware goes through the `CAMRADriver`.

`#include <camra_driver.h>`

Public Member Functions

- **CAMRADriver** (const string &appName, CAMRAConfig *config)
- void **registerHandler** (MessageHandler *h)
- int **start** ()
- void **stop** ()
- const CAMRAConfig & **config** ()
- Exposure & exposure (void)
- void stash_temps (void)
- void reset_tcs (void)
- void populate_onewire_device_strings (void)

Remote methods

Methods exposed through the Receiver. These map to Handler instances declared in the application's `main()` driver.

- void **turn_off_leds** (int cntl_id)
- void **set_ccd_temp** (int ccd_id, double temp)
- void **set_ccd_temp_xfer** (int ccd_id, double bias, double gain)
- void **set_cryo_temp_xfer** (int cntl_id, double bias, double gain)
- void **set_bias_offset** (int ccd_id, const string &amp, int offset)
- void **set_clock_voltages** (int ccd_id, const std::map< string, float > voltages)
- void **abort** (bool write_data=false)
- void **open_shutter** (void)
- void **close_shutter** (void)
- void **enable_shutter** (void)
- void **disable_shutter** (void)
- void **enable_pressure_gauges** (void)
- void **disable_pressure_gauges** (void)
- void **reset_clock_voltages** (void)
- void **reset** (void)
- void **reset_mux** (int index)
- void **override_ifu_associations** (int cntl, string ifu_id, int ifu_slot, string spectrograph_slot)
- Message * **expose** (const ExposureDescriptor &d)
- Message * **get_exposure_info** (void)
- Message * **get_ccd_temp** (int id, bool update_state=false)
- Message * **get_cryo_temp** (int id, bool update_state=false)
- Message * **get_cryo_pressure** (int id, bool update_state=false)
- Message * **get_heater_voltage** (int id, bool update_state=false)
- Message * **get_hardware_status** (bool update_state=false)
- Message * **get_clock_voltages** (void)
- Message * **command_controller** (int id, int command, int arg1, int arg2, int arg3, int arg4)

9.89.2 Member Function Documentation

9.89.2.1 Exposure & CAMRADriver::exposure (void)

Returns a reference to the current active exposure. Throws an exception if there isn't one.

9.89.2.2 void CAMRADriver::populate_onewire_device_strings (void)

Reads and parses onewire device strings from the entire array.

9.89.2.3 void CAMRADriver::reset_tcs (void)

Resets the connection to TCS server.

9.89.2.4 void CAMRADriver::stash_temps (void)

Writes the current temperature setpoints to disk to be read on the subsequent startup.

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

- camra/camra_driver.h
- camra/camra_driver.cpp

9.90 CAMRAException Class Reference

Public Member Functions

- **CAMRAException** (tcsexcept::tcs_exception_t __camra, const std::string &w, const char ∗file, int line, const char ∗func)

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

- het_lib/common/include/tcs_exception.h

9.91 camra_handlers::CAMRAHandler Class Reference

Base class for all remote interfaces.

9.91.1 Detailed Description

The CAMRAHandler base class provides common functionality used by all CAMRA Handlers.

`#include <camra_handlers.h>`

Public Member Functions

- **CAMRAHandler** (CAMRADriver *d, const MessageHandlerDescriptor &m)

Public Attributes

- CAMRADriver * driver

  Pointer binding the handler to an instance of the CAMRADriver.

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

- camra/camra_handlers.h

9.92 CAMRAHardware Class Reference

Hardware encapsulation class.

9.92.1 Detailed Description

Root element of the hardware tree.

`#include <camra_hardware.h>`

Public Member Functions

- **CAMRAHardware** (const CAMRAConfig &config)
- bool **update_status** (void)
- void **broadcast_status** (void)
- void **build_status_payload** (Payload &p)
- void **start_expose** (const ExposureDescriptor &descriptor)
- void **end_expose** (double &dark_time, tthread::Timer &read_timer)
- void pivot_from_arc (ExposureDescriptor &descriptor)

  Moves data from the given buffer set into the CAMRAHardware object.

- void fill_with_test_pattern (int rows, int cols)

  Fills all data regions with a test pattern.

- void **set_global_ccd_temperature** (double temp)
- void **set_ccd_temperature** (int id, double temp, bool lock=true)
- void **turn_off_leds** (int id)
- double **get_ccd_temp** (int id, bool update_state=false)
- double **get_heater_voltage** (int id, bool update_state=false)
- double **get_cryo_temp** (int id, bool update_state=false)
- double **get_cryo_pressure** (int id, bool update_state=false)
- void **set_binning** (unsigned y_binning, unsigned x_binning)
- void **set_bias_offset** (int id, const string &amp, int count)
- void **set_clock_voltages** (int ccd_id, std::map< string, float > &voltages)
- void **reset_clock_voltages** (void)
- void **get_clock_voltages** (Payload &p)
- map< int, Cntl * > & **cntls** (void)
- map< int, CCD * > & **ccds** (void)
- list< Amp * > & **amps** (void)
- unsigned **valid_ccd_count** (void)
- void **enable_pressure_gauges** (void)
- void **disable_pressure_gauges** (void)
- void **check_for_light** (bool ignore_exception=false)
- void **store_last_known_temps** (void)
- string **command_controller** (int id, int command, int arg1, int arg2, int arg3, int arg4)
- void **reset_readout_buffer** (void)
- void **reset_mux** (int index)
- void **populate_onewire_device_strings** (void)
- void **set_ccd_temp_xfer** (int ccd_id, double bias, double gain)
- void **set_cryo_temp_xfer** (int cntl_id, double bias, double gain)
- void **override_ifu_associations** (int cntl, string ifu_id, int ifu_slot, string spectrograph_slot)

Protected Member Functions

- void **apply_bias_offset** (int count, int ccd=0, const string &="all")
- void **apply_clock_voltages** (int ccd_id, double s_hi_1, double s_lo_1, double p_hi_1, double p_-lo_1, double s_hi_2, double s_lo_2, double p_hi_2, double p_lo_2, double s_hi_3, double s_lo_3, double p_hi_3, double p_lo_3, double rg_hi, double rg_lo, double sw_hi, double sw_lo, double lower_od, double upper_od, double lower_pa, double upper_pa, double og, double rd)
- void **enumerate_controllers** (arc::CArcMuxPCIe *mux_ptr, int mux_count)
- void **expose** (void)
- virtual void **write** (FitsOutWrapper &f)
- void **build_default_hardware** (const CAMRAConfig &config)
- void **apply_last_known_temps** (void)

Protected Attributes

- const CAMRAConfig & **camra_config_**
- tthread::fast_mutex **exposure_mutex_**
- tthread::mutex **shutter_mutex_**
- tthread::mutex **readout_mutex_**
- tthread::condition_variable **shutter_cond_**
- tthread::condition_variable **readout_cond_**
- arc::CArcSys **arc_**
- tthread::mutex **mutex_**
- bool **debug_**
- list< Amp * > **amps_**
- map< int, CCD * > **ccds_**
- map< int, Cntl * > **cntls_**
- map< int, Mux * > **muxes_**
- tthread::ThreadWorker * **status_thread_**
- string **output_dir_**
- string **microcod_str_**
- string **mcodmd5_str_**
- unsigned **x_binning_**
- unsigned **y_binning_**
- unsigned **x_pixels_**
- unsigned **y_pixels_**
- unsigned **valid_ccd_count_**
- std::list< bool > **controller_validity_flags_**
- std::list< int > **controller_ids_**
- std::list< int > **ifu_slots_**
- tthread::thread * **exposure_thread_**
- ExposureDescriptor & **descriptor_**
- tthread::Timer **dark_timer_**
- TCSBaseException **error_**
- volatile bool **exposure_error_**
- volatile bool **gauges_enabled_**
- double **readout_delay_**
- bool **hardware_disabled_**

Friends

- class Mux
- class Cntl
- class CCD
- class Amp

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

- camra/camra_hardware.h
- camra/camra_hardware.cpp

9.93 cancel_trajectory Class Reference

9.93.1 Detailed Description

Cancels any currently active trajectory.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **cancel_trajectory** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.93.2 Member Function Documentation

9.93.2.1 void cancel_trajectory::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.94 CBUF< indexType, size, entryType > Class Template Reference

template<class indexType, unsigned size, class entryType> class CBUF< indexType, size, entryType >

Public Member Functions

- indexType **Len** () const
- indexType **Size** () const
- bool **IsEmpty** () const
- bool **IsFull** () const
- bool **Error** () const
- void **Push** (entryType val)
- entryType **Pop** ()
- entryType **Sum** ()

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

- gui/include/cameraImageTabUi.h

9.95 geHeaters::Ccd Class Reference

Public Member Functions

- def **__init__**
- def **addData**
- def **match**
- def **whoAmI**

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

- integration/jrf/scripts/geHeaters.py

9.96 CCD Class Reference

Class corresponding to a single CCD.

`#include <camra_hardware.h>`

Public Member Functions

- **CCD** (int idx, int id, int side, const CAMRAConfig &config, Cntl *cntl)
- int side (void)

  Returns the side of the spectrograph the CCD is on, LEFT or RIGHT.

- Cntl & cntl (void)

  Returns reference to the Cntl this CCD is associated with.

- double **get_temperature** (void)
- double **get_heater_voltage** (void)
- void **get_clock_voltages** (Payload &p)

Protected Member Functions

- virtual void **write** (FitsOutWrapper &f)
- void **update_temperature** (int adu)
- void **update_heater_voltage** (double voltage)
- Amp & **upper** (void)
- Amp & **lower** (void)
- void **reset_clock_voltages** (bool do_exception=true)
- void **set_clock_voltages** (std::map< string, float > &voltages)
- void **set_bipolar_voltage** (std::map< string, float > voltages, const string &key, int board_type, int left_pin, int right_pin, double range, int bitdepth)
- void **set_unipolar_voltage** (std::map< string, float > voltages, const string &key, int board_type, int left_pin, int right_pin, double range, int bitdepth)

Protected Attributes

- int **id_**
- Cntl * **cntl_**
- int **side_**
- list< Amp * > **amps_**
- int **amp_config_**
- string **ccdpos_str_**
- double **ttf_bias_**
- double **ttf_gain_**
- double **temperature_**
- double **temperature_setpoint_**
- double **heater_voltage_**
- std::map< string, float > **configured_clock_voltages_**
- std::map< string, float > **effective_clock_voltages_**
- std::map< string, coefs_t > **configured_voltage_correction_coefs_**

Friends

- class CAMRAHardware
- class Mux
- class Cntl
- class Amp

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

- camra/camra_hardware.h
- camra/camra_hardware.cpp

9.97 center Class Reference

9.97.1 Detailed Description

Moves the probe to the center the object within the aperture, based on the current instantaneous error in telecentric x,y as reported by the WFS metrology pipeline.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **center** (WFSLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.97.2 Member Function Documentation

9.97.2.1 void center::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.98 CentroidInfo Class Reference

Public Member Functions

- **CentroidInfo** (uint cenmethod, uint threshmethod, uint niter, uint PSFmodel, float PSFdia, float thresh_percent, float thresh_nsigma)
- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)

Public Attributes

- uint **m_cenmethod**
- uint **m_threshmethod**
- uint **m_niter**
- uint **m_PSFmodel**
- float **m_PSFdia**
- float **m_thresh_percent**
- float **m_thresh_nsigma**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.99 check Class Reference

Public Member Functions

- **check** (IglooSensors &d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.99.1 Member Function Documentation

9.99.1.1 void check::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- igloo_sensors/igloo_sensors.h

9.100 chunkHeader Class Reference

Public Attributes

- char **sGroupID** [4]
- unsigned int **dwChunkSize**

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

- gui/include/guiSound.h

9.101 cJSON Struct Reference

Public Attributes

- struct cJSON * **next**
- struct cJSON * **prev**
- struct cJSON * **child**
- int **type**
- char * **valuestring**
- int **valueint**
- double **valuedouble**
- char * **string**

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

- het_lib/cJSON/include/cJSON.h

9.102 cJSON_Hooks Struct Reference

Public Attributes

- void ∗(∗ **malloc_fn**)(size_t sz)
- void(∗ **free_fn**)(void ∗ptr)

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

- het_lib/cJSON/include/cJSON.h

9.103 CleanerConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- pas::CleaningMode_t **m_cleaningMode**
- string **m_biasType**
- bool **m_bUseFlats**
- bool **m_bUseMasks**
- bool **m_bUseHotPixels**
- bool **m_bUseOverScan**
- pas::CleaningBiasMode_t **m_cleaningBiasMode**
- string **m_flatFieldImageFile** [5]
- string **m_hotPixelsFile**
- string **m_maskDefFile**

9.103.1 Member Function Documentation

9.103.1.1 void CleanerConfig::set_options (OptionParser & *optionParser*) `[virtual]`

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

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

- pas/include/pasConfig.h
- pas/src/pasConfig.cpp

9.104 clear_fiducial Class Reference

9.104.1 Detailed Description

Resets the fiducial of the underlying metrology system so that the next reading becomes the new fiducial.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **clear_fiducial** (GuideLoop *l, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.104.2 Member Function Documentation

9.104.2.1 void clear_fiducial::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.105 clear_trajectories Class Reference

9.105.1 Detailed Description

Cancels any active trajectory and clear both the current and next trajectories.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **clear_trajectories** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.105.2 Member Function Documentation

9.105.2.1 void clear_trajectories::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.106 clearError Class Reference

9.106.1 Detailed Description

Clears a TMCS error by sending the CLR command to TMCS.

Required parameters:

idx<int> - XXX

`#include <tracker_handler_doc.h>`

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

- tracker/tracker_handler_doc.h

9.107 tracker_handlers::clearError Class Reference

Public Member Functions

- **clearError** (TrackerDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.107.1 Member Function Documentation

9.107.1.1 void tracker_handlers::clearError::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tracker/tracker_handlers.h

9.108 Client Class Reference

9.108.1 Detailed Description

XXX

`#include <tracker_socket.h>`

Public Member Functions

- **Client** (const std::string &host, int port)

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

- tracker/tracker_socket.h
- tracker/tracker_socket.cpp

9.109 Close Class Reference

Close the shutter.

9.109.1 Detailed Description

Parameters:

none

`#include <legacyShutterSubSystem.h>`

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

- legacy/include/legacyShutterSubSystem.h

9.110 camra_handlers::close_shutter Class Reference

Public Member Functions

- **close_shutter** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.110.1 Member Function Documentation

9.110.1.1 void camra_handlers::close_shutter::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.111 close_shutter Class Reference

9.111.1 Detailed Description

Close the configured shutter. This may be a TTL driven lab shutter or the PFIP shutter so exercise caution.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.112 CloseShutter Class Reference

9.112.1 Detailed Description

Close the focal plane shutter.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **CloseShutter** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ACQShutterHandlers.cpp

9.113 cmdConfig Class Reference

9.113.1 Detailed Description

Very minimal shell scripting command interface to TCS subsystems. Sends a single tcs command synchronously to a subsystem and prints the result to stdout

Public Member Functions

- **cmdConfig** (bool isNamed=0)

Public Attributes

- string **route**
- bool **send_async_**

Protected Member Functions

- void print_summary (void)

  Called prior to displaying options.

- void set_options (OptionParser &optionParser)

  Adds options to default set.

9.113.2 Member Function Documentation

9.113.2.1 void cmdConfig::print_summary (void) [protected, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.113.2.2 void cmdConfig::set_options (OptionParser & *optionParser*) [protected, virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- tcsconsole/tcscmd.cpp

9.114 Cntl Class Reference

Class corresponding the a single controller.

`#include <camra_hardware.h>`

Public Member Functions

- **Cntl** (int idx, int id, const CAMRAConfig &config, Mux *mux)
- Mux & **mux** ()
- double **get_cryo_temp** (void)
- double **get_cryo_pressure** (void)
- bool **is_cryo_pressure_valid** (void)
- void **schedule_pressure_gauge_reconnect** (void)

Protected Member Functions

- virtual void **write** (FitsOutWrapper &f)
- void **update_cryo_temp** (int adu)
- void **query_cryo_pressure** (void)
- virtual bool **checkReply** (int nCmd, unsigned int replyLength, unsigned char *replyBuf)
- bool **initialize_and_verify_onewire** (const CAMRAConfig &config)
- bool **is_nonstandard_amp_config** (void)
- CCD & **left** (void)
- CCD & **right** (void)
- bool **enable_pressure_gauge** (void)
- void **disable_pressure_gauge** (void)

Protected Attributes

- SpectrographConfig * **config_**
- Mux * **mux_**
- string **flex_**
- map< int, CCD * > **ccds_**
- string **ifu_id_**
- int **ifu_slot_**
- string **cntl_id_str_**
- string **timing_str_**
- string **clock_driver_str_**
- string **video_str_**
- string **power_cntl_str_**
- string **flex_str_**
- string **ifu_slot_str_**
- string **spectrograph_slot_str_**
- int **spectro_id_**
- double **ttf_bias_**
- double **ttf_gain_**
- double **cryo_temp_**
- double **cryo_pressure_**

- volatile bool **cryo_pressure_valid_**
- volatile bool **pressure_gauge_enabled_**
- volatile bool **pressure_gauge_reconnect_requested_**
- int **characterization_controller_id_**

Friends

- class CAMRAHardware
- class Mux
- class CCD
- class Amp

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

- camra/camra_hardware.h
- camra/camra_hardware.cpp

9.115 coefs_t Struct Reference

Public Attributes

- float **coefs** [2]

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

- camra/camra_config.h

9.116 Command Class Reference

Public Member Functions

- **Command** (char *input)
- std::string **handler** ()
- std::string **message** ()
- bool **async** ()
- bool **valid** ()

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

- tcsconsole/command.h

9.117 camra_handlers::command_controller Class Reference

Public Member Functions

- **command_controller** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.117.1 Member Function Documentation

9.117.1.1 void camra_handlers::command_controller::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.118 command_controller Class Reference

9.118.1 Detailed Description

Sends the given command to the specified controller.

Required parameters:

id<int> - Controller ID to operate on.

command<int> - Integer representation of the command to be sent. For example, to send the POF command, give command=0x504F46.

Optional parameters:

arg1<int> - Integer argument to be passed to the requested command.

arg2<int> - Integer argument to be passed to the requested command.

arg3<int> - Integer argument to be passed to the requested command.

arg4<int> - Integer argument to be passed to the requested command.

```
#include <camra_handler_doc.h>
```

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

- camra/camra_handler_doc.h

9.119 tthread::condition_variable Class Reference

9.119.1 Detailed Description

Condition variable class. This is a signalling object for synchronizing the execution flow for several threads. Example usage:

```
// Shared data and associated mutex and condition variable objects
int count;
mutex m;
condition_variable cond;

// Wait for the counter to reach a certain number
void wait_counter(int targetCount)
{
  lock_guard<mutex> guard(m);
  while(count < targetCount)
    cond.wait(m);
}

// Increment the counter, and notify waiting threads
void increment()
{
  lock_guard<mutex> guard(m);
  ++ count;
  cond.notify_all();
}
```

`#include <tinythread.h>`

Public Member Functions

- condition_variable ()
  Constructor.
- ~condition_variable ()
  Destructor.
- template<class _mutexT >
  bool wait (_mutexT &aMutex)
- template<class _mutexT >
  bool wait_for (_mutexT &aMutex, double seconds, double *spent=NULL)
- void notify_one ()
- void notify_all ()

9.119.2 Member Function Documentation

9.119.2.1 void tthread::condition_variable::notify_all () [inline]

Notify all threads that are waiting for the condition. All threads that are blocked waiting for this condition variable will be woken up.

Note:

Only threads that started waiting prior to this call will be woken up.

9.119.2.2 void tthread::condition_variable::notify_one () [inline]

Notify one thread that is waiting for the condition. If at least one thread is blocked waiting for this condition variable, one will be woken up.

Note:

Only threads that started waiting prior to this call will be woken up.

9.119.2.3 template<class _mutexT > bool tthread::condition_variable::wait (_mutexT & *aMutex*) [inline]

Wait for the condition. The function will block the calling thread until the condition variable is woken by `notify_one()` or `notify_all()`.

Parameters:

← ***aMutex*** A mutex that will be unlocked when the wait operation starts, and locked again as soon as the wait operation is finished.

9.119.2.4 template<class _mutexT > bool tthread::condition_variable::wait_for (_mutexT & *aMutex*, double *seconds*, double * *spent* = NULL) [inline]

Wait for the condition for a specified period of `seconds`. The function will block the calling thread until the condition variable is woken by `notify_one()`, `notify_all()`, or the timeout expires. Returns true if a notification occurs prior to the timeout, false if the timeout expires.

Parameters:

← ***aMutex*** A mutex that will be unlocked when the wait operation starts, and locked again as soon as the wait operation is finished.

Wait for the condition variable notify() to have been called or for the pthread call to time out. All other conditions result in this thread continuing to wait.

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

- het_lib/tthread/include/tinythread.h

9.120 Config Class Reference

9.120.1 Detailed Description

Class encapsulating the ICS configuration service.

`#include <config.h>`

Public Member Functions

- void **init** (void)
- void **help** (void)
- void **set** (const std::string &key, const std::string &value)
- std::string **get** (const std::string &key)
- **Config** (bool isNamed=0)

Public Attributes

- std::string **event_system**
- std::string **remote_system**

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

- ics/config.h
- playground/toy_subsystem/toy_subsystem.cpp
- ics/config.cpp

9.121 ConfigHandler Class Reference

Public Member Functions

- **ConfigHandler** (Config &d, const MessageHandlerDescriptor &m)

Public Attributes

- Config & **driver_**

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

- ics/config.h

9.122 tcs_photometry::ConfigureParams Struct Reference

9.122.1 Detailed Description

Configuration parameters for object detection and aperture photometry

`#include <photometry.h>`

Public Member Functions

- **ConfigureParams** (double mag_zero, double detect_sigma, int min_pixels, int max_pixels, double gain, double nan_value, double saturation, int sky_order, int sky_annulus, double kron_aperture, double fixed_aperture)

Public Attributes

- bool **debug_**
- double **exp_time_sec_**
- float **mag_zero_**
- float **illumination_**
- float **detect_sigma_**
- int **min_pixels_**
- int **max_pixels_**
- float **gain_**
- float **nan_value_**
- float **saturation_**
- int **sky_order_**
- int sky_annulus_

  width of sky annulus in pixels (5)

- float kron_aperture_

  Kron factor for kron-aperture photometry (2.5 ... 3.5).

- float fixed_aperture_

  Fixed-sized aperture for fixed aperture photometry.

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

- het_lib/photometry/include/photometry.h

9.123 wfs_photometry::ConfigureParams Struct Reference

Public Member Functions

- **ConfigureParams** (string recMatrixFile, string refListFile, bool bRenewFlag, int pixelsAcrossSubAperture, int pixelsPerSubAperture, int numZernikeModes, int numZernikeModesToReconstruct, double diskX, double diskY, double diskRadius, double pixelScale, double pupilRadius, double rotation, double refWaveLength)

Public Attributes

- bool **m_debug**
- string **m_recMatrixFile**
- string **m_refListFile**
- bool **m_bRenewFlag**
- int **m_pixelsAcrossSubAperture**
- int **m_numZernikeModes**
- int **m_numZernikeModesToReconstruct**
- double **m_diskX**
- double **m_diskY**
- double **m_diskRadius**
- double **m_pixelScale**
- int **m_pixelsPerSubAperture**
- double **m_pupilRadius**
- double **m_rotation**
- double **m_refWaveLength**

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

- het_lib/wfs_lib/include/wfs_photometry.h

9.124 Ui::configureSystemsDialog Class Reference

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

- gui/build/ui_configureSystemsDlg.h

9.125 configureSystemsDialog Class Reference

Protected Member Functions

- void **loadSettings** (void)
- void **saveSettings** (void)

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

- gui/include/configureSystemsDlg.h
- gui/src/configureSystemsDlg.cpp

9.126 configureTabsDialog Class Reference

Protected Member Functions

- void **loadSettings** (void)
- void **saveSettings** (void)
- void **loadSettings** (void)
- void **saveSettings** (void)

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

- gui/include/configureTabsDlg.h
- utility_gui/include/configureTabsDlg.h
- gui/src/configureTabsDlg.cpp
- utility_gui/src/configureTabsDlg.cpp

9.127 Ui::configureTabsDialog Class Reference

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

- gui/build/ui_configureTabsDlg.h

9.128 confirmOffsetDialog Class Reference

Public Member Functions

- **confirmOffsetDialog** (cameraImageScene *pParent)

Protected Member Functions

- void **loadSettings** (void)
- void **saveSettings** (void)

Protected Attributes

- cameraImageScene * **m_pParent**

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

- gui/include/cameraImageScene.h
- gui/src/cameraImageScene.cpp

9.129 Ui::confirmOffsetDialog Class Reference

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

- gui/build/ui_confirmOffsetDlg.h

9.130 consoleConfig Class Reference

Public Member Functions

- **consoleConfig** (bool isNamed=0)

Public Attributes

- string **route**
- string **runtimeRoute**
- bool **exit_on_error**

Protected Member Functions

- void print_summary (void)

  Called prior to displaying options.

- void set_options (OptionParser &optionParser)

  Adds options to default set.

9.130.1 Member Function Documentation

9.130.1.1 void consoleConfig::print_summary (void) [protected, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.130.1.2 void consoleConfig::set_options (OptionParser & *optionParser*) [protected, virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- tcsconsole/tcsconsole.cpp

9.131 ControllerConfig Class Reference

Public Member Functions

- void set_options (OptionParser &o)

  Adds options to default set.

- void print_summary (void)

  Called prior to displaying options.

Public Attributes

- int controller_id

  Serial number of the controller.

- std::map< string, coefs_t > **ccd_right_voltage_correction_coefs**
- std::map< string, coefs_t > **ccd_left_voltage_correction_coefs**
- float **ccd_left_temperature_transfer_function** [2]
- float **ccd_right_temperature_transfer_function** [2]
- float **cryo_temperature_transfer_function** [2]

9.131.1 Member Function Documentation

9.131.1.1 void ControllerConfig::print_summary (void) [virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.131.1.2 void ControllerConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- camra/camra_config.h
- camra/camra_config.cpp

9.132 CoordinateTransformException Class Reference

Public Member Functions

- **CoordinateTransformException** (tcsexcept::tcs_exception_t __coordinate_transform, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.133 correct_for_structure Class Reference

9.133.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **correct_for_structure** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.133.2 Member Function Documentation

9.133.2.1 void correct_for_structure::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.134 Correction Class Reference

9.134.1 Detailed Description

a single correction to be applied to a trajectory

`#include <metrology.h>`

Public Member Functions

- **Correction** (const std::string &who, double dx, double dy, double dz, double drho, double dtheta, double dphi)
- **Correction** (const std::string &who, const itf_coordinate &ditf)
- **Correction** (const std::string &who, const double &dw)
- **Correction** (const std::string &who, const itf_coordinate &ditf, const double &dw)
- **Correction** (const Correction &c)
- **Correction** (int zero=0)
- Correction **operator+=** (const Correction &c)
- virtual void **apply_parms** (Payload &p) const

Public Attributes

- std::string **who**
- itf_coordinate **ditf**
- double **dw**

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

- tcs/metrology.h
- tcs/metrology.cpp

9.135 cubFit Class Reference

Public Types

- typedef double **value_type**

Public Member Functions

- double **operator()** (const double &x, const ltl::FVector< double, 4 > &p, ltl::FVector< double, 4 > &jac) const

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

- het_lib/coord_trans/mount_model/curveFits.h

9.136 AcqCWFSStatus::cwfs_status Struct Reference

Public Member Functions

- QString **cwfs** (void)
- QString **pv** (void)
- QString **cwfsCameraMirror** (void)
- **BOOL_VAR** (cwfsCameraPower)
- **BOOL_VAR** (cwfsCameraPowerAlert)
- **BOOL_VAR** (pvCameraPower)
- **BOOL_VAR** (pvCameraPowerAlert)

Public Attributes

- int **m_cwfs**
- int **m_pv**
- int **m_cwfsCameraMirror**

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

- gui/include/pfipStatusData.h

9.137 CWFSInfo Class Reference

Public Member Functions

- void **SendCWFSCamInfo** (void)
- void **SetCWFSCameraDeployed** (int cwfsCameraDeployed)
- void **SetPVCameraDeployed** (int pvCameraDeployed)
- void **SetCWFSCameraMirrorPos** (int pos)
- void **SetPV_CWFSCameraPower** (bool power)
- bool **GetPV_CWFSCameraPower** ()

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

- pfip/include/PFIPStatus.h
- pfip/src/EventInfo.cpp

9.138 CyclePipeline Class Reference

9.138.1 Detailed Description

Single cycle the indicated pipeline, The pipeline (camera) name, is prepended to the handler name (i.e. Guider1_).

Required parameters:

procMode<string> - Image processing mode. One of the following strings. 'guiding', 'tip-tilt', 'image-analysis', 'wave-front-sensing'.

imageFilename<string> - Name of file to store the image in (relative the startup directory).

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.139 mainWindow::DashBoardAlerts Class Reference

Public Attributes

- tthread::mutex **m_AccessMutex**
- int **m_WarningCount**
- int **m_ErrorCount**
- int **m_FatalErrorCount**
- int **m_AlarmCount**
- int **m_TempWarningCount**
- int **m_TempErrorCount**
- int **m_TempFatalErrorCount**
- int **m_TempAlarmCount**
- bool **m_bWarning**
- bool **m_bError**
- bool **m_bFatalError**
- bool **m_bAlarm**
- int **m_TabIndex**
- QStringList **m_WarningMsgList**
- QStringList **m_ErrorMsgList**
- QStringList **m_FatalErrorMsgList**
- QStringList **m_AlarmMsgList**
- QString **m_WarningMsgs**
- QString **m_ErrorMsgs**
- QString **m_FatalErrorMsgs**
- QString **m_AlarmMsgs**

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

- gui/include/mainWindow.h

9.140 Payload::Data Class Reference

Public Member Functions

- **Data** (const void ∗data, size_t size)
- const void ∗ **data** (void) const
- size_t **size** (void) const

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

- het_lib/messaging/include/payload.h
- het_lib/messaging/src/payload.cpp

9.141 dataChunk Class Reference

Public Attributes

- chunkHeader **header**
- unsigned char **sampleData** [0]

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

- gui/include/guiSound.h

9.142 deactivate Class Reference

9.142.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **deactivate** (MetrologyLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.142.2 Member Function Documentation

9.142.2.1 void deactivate::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.143 deep_arg_t Struct Reference

Public Attributes

- double **aodp**
- double **cosio**
- double **sinio**
- double **omgdot**
- double **xmdot**
- double **xnodot**
- double **xnodp**
- double **eosq**
- double **betao**
- double **theta2**
- double **sing**
- double **cosg**
- double **betao2**
- double **xll**
- double **omgadf**
- double **xnode**
- double **em**
- double **xinc**
- double **xn**
- double **t**
- double **d2201**
- double **d2211**
- double **d3210**
- double **d3222**
- double **d4410**
- double **d4422**
- double **d5220**
- double **d5232**
- double **d5421**
- double **d5433**
- double **atime**
- double **del1**
- double **del2**
- double **del3**
- double **e3**
- double **ee2**
- double **omegaq**
- double **pe**
- double **pgh**
- double **ph**
- double **pinc**
- double **pl**
- double **preep**
- double **savtsn**
- double **se2**
- double **se3**

- double **sgh2**
- double **sgh3**
- double **sgh4**
- double **sh2**
- double **sh3**
- double **si2**
- double **si3**
- double **sl2**
- double **sl3**
- double **sl4**
- double **sse**
- double **ssg**
- double **ssh**
- double **ssi**
- double **ssl**
- double **thgr**
- double **xfact**
- double **xgh2**
- double **xgh3**
- double **xgh4**
- double **xh2**
- double **xh3**
- double **xi2**
- double **xi3**
- double **xl2**
- double **xl3**
- double **xl4**
- double **xlamo**
- double **xli**
- double **xni**
- double **xnq**
- double **xqncl**
- double **zcosgl**
- double **zcoshl**
- double **zcosil**
- double **zmol**
- double **zmos**
- double **zsingl**
- double **zsinhl**
- double **zsinil**
- int **resonance_flag**
- int **synchronous_flag**

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

- tcs/traj_comp/NORADlib/norad_in.h

9.144 delta_time Struct Reference

UT1 definitions (the deltaUT=UT1-UTC structure).

`#include <point.h>`

Public Attributes

- double **value**
- int **MJD**
- double **x_polar**
- double **y_polar**

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

- integration/sir/Het_trajectory/point.h
- tcs/traj_comp/point.h

9.145 DeployACQCameraMirror Class Reference

9.145.1 Detailed Description

Deploy the Acquisition Camera mirror.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **DeployACQCameraMirror** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ACQCameraHandlers.cpp

9.146 DeployCWFSCamera Class Reference

9.146.1 Detailed Description

Deploy the Calibration Wave Front Sensor and retract the Pupil View Camera.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **DeployCWFSCamera** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/CwfsPvHandlers.cpp

9.147 DeployCWFSCameraMirror Class Reference

9.147.1 Detailed Description

Deploy the Calibration Wave Front Sensor mirror.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **DeployCWFSCameraMirror** (Receiver ∗rcv, ACQ_SubSystem ∗acq)
- virtual void HandlerAction (Message ∗pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message ∗pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/CwfsPvHandlers.cpp

9.148 DeployPVCamera Class Reference

9.148.1 Detailed Description

Deploy the Pupil View camera and retract the Calibration Wave Front Sensor.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **DeployPVCamera** (Receiver ∗rcv, ACQ_SubSystem ∗acq)
- virtual void HandlerAction (Message ∗pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message ∗pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/CwfsPvHandlers.cpp

9.149 detachedTabWindow Class Reference

Signals

- void **closed** ()
- void **closed** ()

Public Member Functions

- **detachedTabWindow** (const QString &title)
- **detachedTabWindow** (const QString &title)

Protected Member Functions

- virtual void **closeEvent** (QCloseEvent ∗)
- virtual void **closeEvent** (QCloseEvent ∗)

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

- gui/include/mainWindow.h
- utility_gui/include/mainWindow.h
- gui/build/moc_mainWindow.cpp

9.150 DeviceConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- string **DMI_address**
- string **SAMS_address**
- bool **m_disableSAC**
- bool **m_disableCCAS**

9.150.1 Member Function Documentation

9.150.1.1 void DeviceConfig::set_options (OptionParser & *optionParser*) `[virtual]`

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

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

- pas/include/pasConfig.h
- pas/src/pasConfig.cpp

9.151 DevicePipelineEnableCCASDMIHandlerClass Class Reference

Public Member Functions

- **DevicePipelineEnableCCASDMIHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.151.1 Member Function Documentation

9.151.1.1 virtual void DevicePipelineEnableCCASDMIHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasDevicePipelineSubSystem.h

9.152 DevicePipelineEnableSACDMIHandlerClass Class Reference

Public Member Functions

- **DevicePipelineEnableSACDMIHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.152.1 Member Function Documentation

9.152.1.1 virtual void DevicePipelineEnableSACDMIHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasDevicePipelineSubSystem.h

9.153 DevicePipelineSetGROCHandlerClass Class Reference

Public Member Functions

- **DevicePipelineSetGROCHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.153.1 Member Function Documentation

9.153.1.1 virtual void DevicePipelineSetGROCHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasDevicePipelineSubSystem.h

9.154 DevicePipelineStatusHandlerClass Class Reference

Public Member Functions

- **DevicePipelineStatusHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.154.1 Member Function Documentation

9.154.1.1 virtual void DevicePipelineStatusHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasDevicePipelineSubSystem.h

9.155 camra_handlers::disable_pressure_gauges Class Reference

Public Member Functions

- **disable_pressure_gauges** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.155.1 Member Function Documentation

9.155.1.1 void camra_handlers::disable_pressure_gauges::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.156 disable_pressure_gauges Class Reference

9.156.1 Detailed Description

Disable all cryostat pressure gauges configured in the array.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.157 disable_rho Class Reference

9.157.1 Detailed Description

Forces the rho value to remain fixed at zero the current trajectory points. Note that this only impact trajectory points, not static tracker moves. This interface is for engineering purposes only.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **disable_rho** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.157.2 Member Function Documentation

9.157.2.1 void disable_rho::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.158 camra_handlers::disable_shutter Class Reference

Public Member Functions

- **disable_shutter** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.158.1 Member Function Documentation

9.158.1.1 void camra_handlers::disable_shutter::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.159 disable_shutter Class Reference

9.159.1 Detailed Description

Disable the configured shutter. This may be a TTL driven lab shutter or the PFIP shutter so exercise caution.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.160 DisableProbes Class Reference

9.160.1 Detailed Description

Disables all the probes. If one or more of the probes is disabled by configuration, that probe will not be affected. The PFIPInterrupt() handler can be used to cancel this operation.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **DisableProbes** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.161 DMI_get_status Class Reference

9.161.1 Detailed Description

Returns the current status of the instrument. Throws an exception if the status is unknown. XXX document info returned as it evolves.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **DMI_get_status** (DMILoop *l, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.161.2 Member Function Documentation

9.161.2.1 void DMI_get_status::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.162 pasDevicePipelineSubSystem::DMIData Struct Reference

Public Member Functions

- **DMIData** (DMIData &other)
- bool **operator==** (DMIData &rhs)
- bool **operator!=** (DMIData &rhs)

Public Attributes

- bool **CCASDisabled**
- double **CCASDistance**
- double **CCASLastValidDistance**
- double **CCASPeakAmpl**
- double **CCASCoupl1**
- bool **SACDisabled**
- double **SACDistance**
- double **SACLastValidDistance**
- double **SACPeakAmpl**
- double **SACCoupl1**

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

- pas/include/pasDevicePipelineSubSystem.h

9.163 DMIData Class Reference

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- struct DMIData::status **m_status**

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

- gui/include/pasStatusData.h
- gui/src/pasStatusData.cpp

9.164 DMIHandler Class Reference

Public Member Functions

- **DMIHandler** (DMILoop *l, const MessageHandlerDescriptor &m)

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

- tcs/metrology.h

9.165 DMILoop Class Reference

Public Member Functions

- **DMILoop** (Instrument &instrument, const TCSServerConfig &config, TCSDriver *driver, Metrology *master=NULL)
- void **start** (Message &m)
- void **start** (void)
- void **activate** (void)
- void **set_focus** (const double &w, bool global=0)
- double **get_latest_distance** (void)
- double **get_focus** (void)
- Message * **get_status** (void)
- virtual void **apply_metrology** (Payload &p, const fptime &data_time, bool do_throw=0)
- virtual void **clear** (bool have_lock=0)

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

- tcs/metrology.h
- tcs/metrology.cpp

9.166 liborchestrate::DmiSubsystem Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.167 Ui::dmiTab Class Reference

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

- gui/build/ui_dmiTab.h

9.168 dmiTabUi Class Reference

Signals

- void **ForceHeartbeat** (std::string)
- void **ForceLoopStateEvent** (std::string)
- void **ForcePipelineStatusEvent** (std::string)
- void **StartPipeline** (std::string)
- void **StopPipeline** (std::string)
- void **StartMetrologyLoop** (std::string)
- void **StopMetrologyLoop** (std::string)
- void **ActivateMetrologyLoop** (std::string)
- void **DeactivateMetrologyLoop** (std::string)
- void **SetFocus** (double)
- void **SetIntervalSpeed** (std::string, double)
- void **EnableSACDMIMeasurements** (std::string, bool)
- void **EnableCCASDMIMeasurements** (std::string, bool)

Public Member Functions

- **dmiTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

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

- gui/include/dmiTabUi.h
- gui/build/moc_dmiTabUi.cpp
- gui/src/dmiTabUi.cpp

9.169 Doh Class Reference

Public Member Functions

- const char * **what** () const throw ()

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

- playground/try_catch/try_catch.cpp

9.170 dome_Abort Class Reference

9.170.1 Detailed Description

Abort (hard stop) dome movement.

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.171 dome_GetPosition Class Reference

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

- legacy/include/legacy_handler_doc.h

9.172 dome_Home Class Reference

9.172.1 Detailed Description

Home the dome (69.0).

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.173 dome_IssuePMACCmd Class Reference

9.173.1 Detailed Description

Required parameters:

cmd<string> -

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.174 dome_MoveTo Class Reference

9.174.1 Detailed Description

Required parameters:

target<double> -

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.175 dome_MoveToCCAS Class Reference

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

- legacy/include/legacy_handler_doc.h

9.176 dome_OnLine Class Reference

9.176.1 Detailed Description

Return dome on-line status.

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.177 dome_Reset Class Reference

9.177.1 Detailed Description

Reset dome power inverter.

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.178 dome_SetDirection Class Reference

9.178.1 Detailed Description

Required parameters:

direction<string> -

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.179 dome_SetPower Class Reference

9.179.1 Detailed Description

Required parameters:

state<string> -

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.180 dome_SetVelocity Class Reference

9.180.1 Detailed Description

Required parameters:

velocity<string> -

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.181 dome_Status Class Reference

9.181.1 Detailed Description

Optional parameters:

update_state<bool> -

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.182 dome_Stop Class Reference

9.182.1 Detailed Description

Stop (soft stop) dome movement.

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.183 dome_Stow Class Reference

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

- legacy/include/legacy_handler_doc.h

9.184 dome_Test Class Reference

9.184.1 Detailed Description

Required parameters:

state<string> -

`#include <legacy_handler_doc.h>`

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

- legacy/include/legacy_handler_doc.h

9.185 DomeClient Class Reference

Public Member Functions

- **DomeClient** (const std::string &route)
- void **move** (double az)
- void **wait** (void)

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

- tcs/dome_client.h

9.186 DomeConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- double m_noMoveNeededDistanceDegrees

  No move needed distance.

- double m_positionAccuracyDegrees

  On target position accuracy.

- double **m_moveStartedTimeout**
- double **m_moveStoppingTimeout**
- double **m_moveExtraTimeout**

9.186.1 Member Function Documentation

9.186.1.1 void DomeConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

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

- legacy/include/legacyConfig.h
- legacy/src/legacyConfig.cpp

9.187 DomeStatus Class Reference

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- struct DomeStatus::status **m_status**

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

- gui/include/legacyStatusData.h
- gui/src/legacyStatusData.cpp

9.188 DomeStatusHandlerClass Class Reference

Public Member Functions

- **DomeStatusHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.188.1 Member Function Documentation

9.188.1.1 virtual void DomeStatusHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- legacy/include/legacyDomeSubSystem.h

9.189 Ui::domeTab Class Reference

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

- gui/build/ui_domeTab.h

9.190 domeTabUi Class Reference

Signals

- void **ForceHeartbeat** (std::string)
- void **ForceLegacyStatusEvents** (void)
- void **ForceLoopStateEvent** (std::string)
- void **MoveDomeTo** (double)
- void **SetDomeVelocityTo** (std::string)
- void **SetDomeDirTo** (std::string)

Public Member Functions

- **domeTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Protected Member Functions

- void **LoadVelocityValues** (void)

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

- gui/include/domeTabUi.h
- gui/build/moc_domeTabUi.cpp
- gui/src/domeTabUi.cpp

9.191 droppablePlainTextEdit Class Reference

Public Member Functions

- **droppablePlainTextEdit** (QWidget ∗pParent=NULL)
- bool **canInsertFromMimeData** (const QMimeData ∗source) const
- void **insertFromMimeData** (const QMimeData ∗source)
- **droppablePlainTextEdit** (QWidget ∗pParent=NULL)
- bool **canInsertFromMimeData** (const QMimeData ∗source) const
- void **insertFromMimeData** (const QMimeData ∗source)

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

- gui/include/droppablePlainTextEdit.h
- utility_gui/include/droppablePlainTextEdit.h
- gui/src/droppablePlainTextEdit.cpp
- utility_gui/src/droppablePlainTextEdit.cpp

9.192 tthread::chrono::duration< _Rep, _Period > Class Template Reference

9.192.1 Detailed Description

template<class _Rep, class _Period = ratio<1>> class tthread::chrono::duration< _Rep, _Period >

Duration template class. This class provides enough functionality to implement `this_thread::sleep_for()`.

`#include <tinythread.h>`

Public Types

- typedef _Rep **rep**
- typedef _Period **period**

Public Member Functions

- template<class _Rep2 >
  duration (const _Rep2 &r)

  Construct a duration object with the given duration.

- rep count () const

  Return the value of the duration object.

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

- het_lib/tthread/include/tinythread.h

9.193 DynamicConfig Struct Reference

Public Member Functions

- void **operator=** (DynamicConfig &rhs)
- void **operator=** (DynamicConfig &rhs)

Public Attributes

- bool **itf_data_valid**
- float **itf_x**
- float **itf_y**
- float **itf_z**
- float **itf_rho**
- float **itf_theta**
- float **itf_phi**
- bool **probe_data_valid**
- float **probe_arm_angle**
- float **probe_carriage_angle**
- float **probe_tele_x**
- float **probe_tele_y**
- bool **dmi_data_valid**
- float **dmi**
- double **unixtime**
- std::string **cenoption**
- bool **complete**

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

- het_lib/wfs_lib/include/reducewfs.h
- het_lib/wfs_lib/include/wfslib_het.h

9.194 editPreferencesDialog Class Reference

Public Member Functions

- **editPreferencesDialog** (guiPreferences *pPreferences, QWidget *parent=0)

Protected Member Functions

- void **loadSettings** (void)
- void **saveSettings** (void)

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

- gui/include/editPreferencesDlg.h
- gui/src/editPreferencesDlg.cpp

9.195 Ui::editPreferencesDialog Class Reference

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

- gui/build/ui_editPreferencesDlg.h

9.196 enable_pressure_gauges Class Reference

9.196.1 Detailed Description

Enables all cryostat pressure gauges configured in the array.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.197 camra_handlers::enable_pressure_gauges Class Reference

Public Member Functions

- **enable_pressure_gauges** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.197.1 Member Function Documentation

9.197.1.1 void camra_handlers::enable_pressure_gauges::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.198 enable_rho Class Reference

9.198.1 Detailed Description

Reenables the application the rho value to trajectory points. Note that this only impact trajectory points, not static tracker moves. This interface is for engineering purposes only.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **enable_rho** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.198.2 Member Function Documentation

9.198.2.1 void enable_rho::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.199 camra_handlers::enable_shutter Class Reference

Public Member Functions

- **enable_shutter** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.199.1 Member Function Documentation

9.199.1.1 void camra_handlers::enable_shutter::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.200 enable_shutter Class Reference

9.200.1 Detailed Description

Enable the configured shutter. This may be a TTL driven lab shutter or the PFIP shutter so exercise caution.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.201 EnableCCASDMI Class Reference

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

- pas/include/pas_handler_doc.h

9.202 EnableProbes Class Reference

9.202.1 Detailed Description

Enables all the probes. If one or more of the probes is disabled by configuration, that probe will not be affected. The PFIPInterrupt() handler can be used to cancel this operation.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **EnableProbes** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.203 EnableSACDMI Class Reference

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

- pas/include/pas_handler_doc.h

9.204 EndRangeProbes Class Reference

9.204.1 Detailed Description

Move all the probes and their arms to the end of their respective ranges. If one or more of the probes is diabled by configuration, that probe will not be affected. The PFIPInterrupt() handler can be used to cancel this operation.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **EndRangeProbes** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.205 Enum Class Reference

Public Member Functions

- **Enum** (const std::string &name, const std::string &field_names)
- void **set** (const void *arg)
- virtual void **set** (const unsigned nfields, const void *arg)
- virtual void **apply_parms** (Payload &p) const

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

- epics_relay/array_types.h
- epics_relay/array_types.cpp

9.206 AcqENWAStatus::enwa_status Struct Reference

Public Member Functions

- **INT_VAR** (encoder, 6)
- **INT_VAR** (pos, 6)

Public Attributes

- flags **m_flags**

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

- gui/include/pfipStatusData.h

9.207 ENWAInfo Class Reference

Public Member Functions

- void **SetENWAEncoder** (int enc)
- void **SetENWAStatus** (MAXONDriveStatus &s)
- void **SetENWAFilterPosition** (int pos)
- void **SendENWAInfo** ()
- int **GetENWAFilterPosition** ()
- std::string **GetENWAFilterName** (void)

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

- pfip/include/PFIPStatus.h
- pfip/src/EventInfo.cpp
- pfip/src/PFIPStatus.cpp

9.208 Ephemeris Struct Reference

9.208.1 Detailed Description

Info computed for a satellite and is for a given point on its trajectory.

`#include <point.h>`

Public Attributes

- double **app_ra**
- double **app_dec**
- double **mean_ra**
- double **mean_dec**
- double **obs_zd**
- double **obs_az**
- double **utc**

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

- integration/sir/Het_trajectory/point.h
- tcs/traj_comp/point.h

9.209 EphList Struct Reference

Contains all the information about a satellite's track.

`#include <point.h>`

Public Attributes

- std::string **name**
- int **MJD**
- std::list< Ephemeris > **Eph**

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

- integration/sir/Het_trajectory/point.h
- tcs/traj_comp/point.h

9.210 EphRec Struct Reference

Records read from file for in order to compute a non sideral tracks.

`#include <point.h>`

Public Attributes

- std::vector< double > **mjd**
- std::vector< double > **ra**
- std::vector< double > **dec**

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

- integration/sir/Het_trajectory/point.h
- tcs/traj_comp/point.h

9.211 EPICSComm Class Reference

9.211.1 Detailed Description

A class to manage EPICS communication

`#include <EPICSComm.h>`

Public Member Functions

- EPICSComm (const string &sourceName, const std::string &EPICS_ipAddress, double timeOut)
- bool Connect ()

  Connect to a EPICS with array of variables to manage.

- void **LockEPICS** ()
- void **UnlockEPICS** ()
- int getValues (pfip::EPICSVariableLookupKey varKey, const uint numVals, void *const Value, bool synchronous=1)

  Retrieve an epics variable value.

- int setValues (pfip::EPICSVariableLookupKey varKey, const uint numValues, void *const Values, bool synchronous=1)

  Set value for an epics variable.

- int sendCommand ()

  Send a series of epics variables in one call.

- void close ()

  End the connection - free channels.

- bool **AttachContext** ()
- bool GetString (pfip::EPICSVariableLookupKey varKey, string &value)

  attach another thread to the same EPICS context

- bool **GetInt** (pfip::EPICSVariableLookupKey varKey, int *Value)
- bool **SetString** (pfip::EPICSVariableLookupKey varKey, string &value)
- bool SetInt (pfip::EPICSVariableLookupKey varKey, const int setValue, bool bLockEPICS=1)

  Set an epics variable integer value.

- bool SetFlag (pfip::EPICSVariableLookupKey varKey, const bool setValue)

  Set an epics variable boolean value.

- bool GetFlag (pfip::EPICSVariableLookupKey varKey, bool &Value)

  Retrieve an epics variable boolean value.

- bool SetAndWaitInt (pfip::EPICSVariableLookupKey varKey_1_ID, pfip::EPICSVariableLookupKey varKey_2_ID, const int setValue, const int compValue, const fptime &timeOut)

  Set an epics value ch1 to setValue, wait for ch2 to change to compValue with timeout seconds timeOut.

- bool **SetIntAndWaitBit** (pfip::EPICSVariableLookupKey varKey_1_ID, pfip::EPICSVariableLookupKey varKey_2_ID, const int setValue, const unsigned int compValue, const fptime &timeOut)
- bool **SetAndWaitFlag** (pfip::EPICSVariableLookupKey varKey_1_ID, pfip::EPICSVariableLookupKey varKey_2_ID, const bool newValue, const bool compVal, const fptime &timeOut)

9.211.2 Constructor & Destructor Documentation

9.211.2.1 EPICSComm::EPICSComm (const string & *sourceName*, const std::string & *EPICS_ipAddress*, double *timeOut*)

Represents the an EPICS channel value and data type

9.211.3 Member Function Documentation

9.211.3.1 bool EPICSComm::Connect ()

Parameters:

varArray[in] - array of channels used for this epics connection

9.211.3.2 bool EPICSComm::GetString (pfip::EPICSVariableLookupKey *varKey*, string & *value*)

Retrieve an epics variable integer value (not an array)

Parameters:

ch[in] - the EPICS channel

Value[out] - ptr to place to store the return value

9.211.3.3 int EPICSComm::getValues (pfip::EPICSVariableLookupKey *varKey*, const uint *numVals*, void *const *Value*, bool *synchronous* = 1)

Parameters:

ValueID[in] - ths channelID of the variable

numVals[in] - > 1 if the variable is an array

Value[out] - ptr to place to put values. Provide additional space if variable is an array

synchronous[in] - true if we are waiting for the EPICS I/O to occur before returning

9.211.3.4 bool EPICSComm::SetInt (pfip::EPICSVariableLookupKey *varKey*, const int *setValue*, bool *bLockEPICS* = 1)

Parameters:

ch[in] - the EPICS channel

setValue - the value

9.211.3.5 int EPICSComm::setValues (pfip::EPICSVariableLookupKey *varKey*, const uint *numValues*, void *const *const *Values*, bool *synchronous* = 1)

Parameters:

ValueID[in] - the channel id of the variable to be set

numValues[in] - > 1, variable is an array with numValues entries

Values[in] - ptr to array of values to set

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

- pfip/include/EPICSComm.h
- pfip/src/EPICSComm.cpp

9.212 EPICSException Class Reference

Public Member Functions

- **EPICSException** (tcsexcept::tcs_exception_t __epics, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.213 EPICSHandle Class Reference

Public Member Functions

- void **allocate_process_variable** (void)
- void **install_process_variable** (ParmsObject *process_variable)
- void **log_epics_exception** (struct exception_handler_args args)

Static Public Member Functions

- static void **destroy** (void)
- static EPICSHandle & **instance** (const std::string &ioc_namespace="", const std::string &event_key="")
- static void **configure** (EPICSRelayConfig *config)
- static void **enable** (void)
- static void **disable** (void)
- static std::string & **ioc_namespace** (void)

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

- epics_relay/epics_handle.h
- epics_relay/epics_handle.cpp

9.214 EPICSRelayConfig Class Reference

Public Member Functions

- **EPICSRelayConfig** (bool is_named=false)
- void set_options (OptionParser &o)

  Adds options to default set.

- void print_summary (void)

  Called prior to displaying options.

Public Attributes

- string **ioc_namespace_**
- string **ioc_routes_**
- string **ioc_port_**
- float **rate_hertz_**
- string **event_key_**
- std::string **process_variable_names_double_** [128+1]
- std::string **process_variable_names_enum_** [128+1]
- std::string **process_variable_names_int_** [128+1]
- std::string **process_variable_names_long_** [128+1]
- std::string **process_variable_names_string_** [128+1]
- std::string **process_variable_names_bitfield_** [128+1]
- std::string **process_variable_names_long_array_** [128+1]
- std::string **process_variable_names_int_array_** [128+1]

9.214.1 Member Function Documentation

9.214.1.1 void EPICSRelayConfig::print_summary (void) `[virtual]`

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.214.1.2 void EPICSRelayConfig::set_options (OptionParser & *optionParser*) `[virtual]`

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- epics_relay/config.h
- epics_relay/config.cpp

9.215 EPICSRelayDriver Class Reference

Public Member Functions

- **EPICSRelayDriver** (const string &appName, EPICSRelayConfig *config)
- int **start** ()
- void **stop** ()

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

- epics_relay/driver.h
- epics_relay/driver.cpp

9.216 eq_coords Class Reference

Public Member Functions

- **eq_coords** (double ra, double dec, double equinox)

Public Attributes

- double **ra_**
- double **dec_**
- double **eq_**

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

- het_lib/common/include/tcs_coordinates.h

9.217 Event Class Reference

9.217.1 Detailed Description

Event class that extends a Payload with hierarchical keying of the entire Payload. The key value represents the system that originated the Event and the and key values allow the caller to associate two levels of hierarchy with the Event. The first of these two levels is determined by the associated EventSource instance and the second and free string parameter given at Event instantiation. This triple allows an implementation of the EventSink to filter Events received based on some combination of , , and .

`#include <event.h>`

Public Member Functions

- **Event** (Message &message)
- std::string variable (void)

- template<typename T >
  Event & operator<< (const std::pair< std::string, T > &key_value)
- template<typename T >
  Event & **operator**<< (const std::pair< char *, T > &key_value)
- template<typename T >
  Event & **operator**<< (const std::pair< const char *, T > &key_value)
- Event & **operator**<< (Payload &p)
- Event & **operator**<< (const ParmsObject &p)
- void operator<< (const event_end_t end)
- void **broadcast** (const bool preserve_metadata=false)

Friends

- class EventHeader
- class EventSource
- class EventSink
- class Tap

9.217.2 Member Function Documentation

9.217.2.1 void Event::operator<< (const event_end_t *end*)

Broadcasts the Event on the route associated with the EventSourceRoot.

9.217.2.2 template<typename T > Event& Event::operator<< (const std::pair< std::string, T > & *key_value*) [inline]

Adds the given templated parameter to the Event. A reference to the calling Event is returned.

Reimplemented from Payload.

9.217.2.3 string Event::variable (void)

Returns a pretty-printed string of the , , and key values.

i.e., system.source.key

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

- het_lib/messaging/include/event.h
- het_lib/messaging/src/event.cpp

9.218 replay::EventBuffer Class Reference

Public Member Functions

- def **__init__**
- def **getStartTime**
- def **getNextEvent**

Public Attributes

- **conn**
- **cursor**
- **use_rowfile**
- **row_buffer**
- **forceSort**
- **sort_by**
- **rowfile**
- **rowfile_name**
- **query_result**

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

- webmonitor/replay.py

9.219 eventCmd Class Reference

Public Member Functions

- **eventCmd** (char ∗input)
- std::string **eventSource** ()
- std::string **eventName** ()
- std::string **message** ()
- bool **valid** ()

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

- tcsconsole/tcsevents.cpp

9.220 EventHandler Class Reference

9.220.1 Detailed Description

The EventHandler provides an interface for retrieving event contents on whatever time constraints the application requires. The state associated with each event is updated internal to the EventHandler each time that event is received, with an effective polling interval of the EVENT_MIN_POLL_MSECS defined above.

See the documentation of value() for how to retrieve state form the EventHandler.

`#include <event.h>`

Public Member Functions

- **EventHandler** (const std::string &endpoint)
- void add_system (const std::string &endpoint)
- template<typename T >
  bool value (const std::string &key, const std::string &parm, T &value, double max_age=0, double wait_allowed=-1)
- bool wait (const std::string &key, const double timeout=0, double *spent=NULL)
- void set_global_defaults (bool default_bool, int default_int, unsigned default_uint, fptime default_fptime, double default_double, float default_float, std::string default_string)
- void set_default (bool value, const std::string &key, const std::string &parm)
- void **set_default** (int value, const std::string &key, const std::string &parm)
- void **set_default** (unsigned value, const std::string &key, const std::string &parm)
- void **set_default** (fptime value, const std::string &key, const std::string &parm)
- void **set_default** (double value, const std::string &key, const std::string &parm)
- void **set_default** (float value, const std::string &key, const std::string &parm)
- void **set_default** (std::string value, const std::string &key, const std::string &parm)
- void **set_default** (const char *value, const std::string &key, const std::string &parm)

9.220.2 Member Function Documentation

9.220.2.1 void EventHandler::add_system (const std::string & *endpoint*)

Adds the implied EventSink to the EventHandler and starts processing the events from that endpoint in addition to any previously registered endpoints.

9.220.2.2 void EventHandler::set_default (bool *value*, const std::string & *key*, const std::string & *parm*)

Method with the same effect as set_global_defaults() but defaulting based on specific key/parm combinations. Subsequent calls with the same pair override the current default and defaults set via this method take precedence over any global defaults set.

9.220.2.3 void EventHandler::set_global_defaults (bool *default_bool*, int *default_int*, unsigned *default_uint*, fptime *default_fptime*, double *default_double*, float *default_float*, std::string *default_string*)

Overrides the exception thrown by value when the max_age is exceeded by defaulting values to the given set of defaults.

9.220.2.4 template<typename T > bool EventHandler::value (const std::string & *key*, const std::string & *parm*, T & *value*, double *max_age* = 0, double *wait_allowed* = -1) [inline]

Reads the parameter, given by parm, from the event, given by key, and writes it to value.

If max_age is positive and the requested event has been received within the allowed max_age, then value is updated with the value from that event and the method returns true. If the requested event has not been received within the allowed max_age, then the method blocks for wait_allowed seconds waiting on the next event. If wait_allowed less than zero, then the method waits indefinitely.

If wait_allowed expires without a timely event, then false is returned. If set_defaults() has been called then value will get the default corresponding to the requested data type. If not, then value remains untouched.

The default behavior, with max_age of zero and wait_allowed of zero, is to block indefinitely for the requested event. The age of the event is determined by its reported data time.

Example:

Read the itf.x position of the tracker from the next tcs.tracker.position event received and wait at most 5 seconds.

double x_itf = -9999; bool rc = value("tcs.tracker.position", "itf.x", 0, 5);

Now read the y position, but modify the max age and wait so that we are using the data from the same event that just gave us x_itf. Note that this is just one of many possible idioms.

double y_itf = -9999; rc &= value("tcs.tracker.position", "itf.z", 5, 0);

9.220.2.5 bool EventHandler::wait (const std::string & *key*, const double *timeout* = 0, double * *spent* = NULL)

Returns immediately if timeout is zero and the requested event has already been received. Otherwise, when timeout is zero, blocks indefinitely for the next requested event. If timeout is non-zero, then the event is waited on for the requested number of seconds.

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

- het_lib/messaging/include/event.h
- het_lib/messaging/src/event.cpp

9.221 EventHeader Class Reference

Public Member Functions

- std::string **system** (void)
- std::string **source** (void)
- std::string **key** (void)

Friends

- class Event
- class Tap

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

- het_lib/messaging/include/event.h
- het_lib/messaging/src/event.cpp

9.222 EventList Class Reference

Public Member Functions

- void **insert** (Event *e)

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

- top/tcs_top_driver.h

9.223 Ui::eventMonitorTab Class Reference

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

- gui/build/ui_eventMonitorTab.h

9.224 eventMonitorTabUi Class Reference

Public Member Functions

- **eventMonitorTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Protected Member Functions

- **DEFINE_RECEIVE_EVENTS_THREAD** (eventMonitorTabUi, EventMonitor)
- void **ReceiveEventMonitorEvents** (void)
- void **loadSettings** (void)
- void **saveSettings** (void)

Protected Attributes

- QStringList **m_systems**
- bool **m_bUpdating**
- QList< QListWidgetItem * > **m_itemList**
- tthread::mutex **m_DBVarsAccessMutex**
- volatile bool **m_bDBLogging**
- LoggerDB * **m_pDBFile**

Friends

- class eventMonitorTabUiThreadWorker

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

- gui/include/eventMonitorTabUi.h
- gui/src/eventMonitorTabUi.cpp

9.225 eventMonitorTabUiThreadWorker Class Reference

Signals

- void **EventReceived** (std::string, std::string)

Public Member Functions

- **eventMonitorTabUiThreadWorker** (const char ∗name, eventMonitorTabUi ∗pParent)
- void **ReceiveEventMonitorEvents** (void)

Protected Attributes

- eventMonitorTabUi ∗ **m_pParent**

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

- gui/include/eventMonitorTabUi.h
- gui/build/moc_eventMonitorTabUi.cpp
- gui/src/eventMonitorTabUi.cpp

9.226 eventsConfig Class Reference

Public Member Functions

- **eventsConfig** (bool isNamed=0)

Public Attributes

- string **system**

Protected Member Functions

- void print_summary (void)

  Called prior to displaying options.

- void set_options (OptionParser &optionParser)

  Adds options to default set.

9.226.1 Member Function Documentation

9.226.1.1 void eventsConfig::print_summary (void) [protected, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.226.1.2 void eventsConfig::set_options (OptionParser & *optionParser*) [protected, virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- tcsconsole/tcsevents.cpp

9.227 EventSink Class Reference

9.227.1 Detailed Description

The EventSink provides an interface to listen for, and optionally filter, broadcast events.

`#include <event.h>`

Public Member Functions

- EventSink (const std::string endpoint="")
- Event * next_event (double timeout_secs=-1.0)
- Event * **next_event** (double timeout_secs, bool &timeout_occurred)
- void add_system (const std::string &endpoint)
- void set_filter (const std::string &system, const std::string &source, const std::string &key)
- void disable (void)

9.227.2 Constructor & Destructor Documentation

9.227.2.1 EventSink::EventSink (const std::string *endpoint* = "")

Constructs an EventSink and connects it to the given endpoint. Events are queued until the next_event() method is called. If no endpoint is given, the sink is initialized but will not receive until add_system() has been called.

9.227.3 Member Function Documentation

9.227.3.1 void EventSink::add_system (const std::string & *endpoint*)

Updates the EventSink to subscribe to an additional system's events.

9.227.3.2 void EventSink::disable (void)

Shuts down the EventSink. This can be used to stop the EventSink from a thread other than the one calling next_event(), in which case, next_event() will return NULL. The EventSink cannot be restarted after a call to disable().

9.227.3.3 Event * EventSink::next_event (double *timeout_secs* = -1.0)

Blocks waiting for an Event and returns the next Event that passes the current filter. If next_event() is called, prior to setting a filter, then the first event received may not match the filter. The caller is responsible for calling delete on the Event* returned when it is no longer needed.

If timeout is 0.0, then polls for at most EVENT_MIN_POLL_MSECS and returns NULL if no event has been received in that poll period.

9.227.3.4 void EventSink::set_filter (const std::string & *system*, const std::string & *source*, const std::string & *key*)

Sets a filter, given by regular expressions, for each level of the event hierarchy.

Listening to all Events corresponds to the ".∗" filter for each parameter.

Throws MessagingException if any of the given regular expressions fails to compile.

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

- het_lib/messaging/include/event.h
- het_lib/messaging/src/event.cpp

9.228 EventSource Class Reference

9.228.1 Detailed Description

The EventSource enables broadcast messaging. Implementations provide a name that will be associated with outgoing Events. An event broadcast is initiated via the EventSource::<< operator and finalized via the Event::<< operator with the event_end_t overload. Ther are no guarantees of delivery and there is no queueing of messages by the EventSource.

`#include <event.h>`

Public Member Functions

- **EventSource** (const std::string &name)
- Event **new_event** (const std::string &key, const fptime timestamp=fptime(0, 0))
- Event **new_event** (const std::string &key, const Payload &p, const fptime timestamp=fptime(0, 0))
- Event * **new_event_ptr** (const std::string &key, const fptime timestamp=fptime(0, 0))
- std::string name (void)

Public Attributes

- event_end_t **ende**

Protected Attributes

- std::string **name_**

9.228.2 Member Function Documentation

9.228.2.1 string EventSource::name (void)

Returns the name of this EventSource given at instantiation.

Reimplemented in ICSConfig, and Instrument.

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

- het_lib/messaging/include/event.h
- het_lib/messaging/src/event.cpp

9.229 EventSourceRoot Class Reference

9.229.1 Detailed Description

Instantiates and configures the EventSourceSocket singleton based on the route and system name given. There can be only one instance of the EventSourceRoot per process. This is the "parent", in terms of the Event keying hierarchy, of all EventSource instantiations.

`#include <event.h>`

Public Member Functions

- **EventSourceRoot** (const std::string &route, const std::string &system)

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

- het_lib/messaging/include/event.h
- het_lib/messaging/src/event.cpp

9.230 EventSourceSocket Class Reference

9.230.1 Detailed Description

Singleton class that wraps the socket used by a process to broadcast Events so that all Events from the process originate from the same IP address.

`#include <event.h>`

Friends

- class Event
- class EventHeader
- class EventSink
- class EventSource
- class EventSourceRoot

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

- het_lib/messaging/include/event.h
- het_lib/messaging/src/event.cpp

9.231 ExportData Class Reference

Public Attributes

- bool **camsetting**
- bool **trksetting**
- bool **wfssetting**
- bool **imgframe**
- bool **wfsfit**
- bool **wfsgrid**
- bool **wfspupil**
- bool **wfsobsc**
- bool **wfsfitmat**
- bool **wfsimgmask**
- bool **wfsppmask**
- string **prefix**
- string **outdir**
- uint **nframe**

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

- het_lib/wfs_lib/include/wfslib.h

9.232 camra_handlers::expose Class Reference

Public Member Functions

- **expose** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.232.1 Member Function Documentation

9.232.1.1 void camra_handlers::expose::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.233 expose Class Reference

9.233.1 Detailed Description

Takes an exposure. The combination of observation, exposure, effective date, and effective instrument must resolve to a unique file name with respect to the files on disk at the point of write. This handler returns when the exposure has begun.

Optional parameters:

object<string> - User chosen name for the \"object\" being observed.

observation<int> - Observation number.

exposure<int> - Exposure number.

dirdate<string> - Directory date. Default is YYYYmmdd based on current UT date.

recipe<string> - The path and filename of the script invoking the command.

subordinate<bool> - Flag indicating whether this exposure is a subordinate in a parallel exposure where the primary shutter is driven by another instrument. This defaults to false.

observer<string> - Name of observer.

operator<string> - Name of operator.

type<string> - The type of the exposure, one of tst, sci, zro, drk, flt, cmp, eng or twi. If given, the type specified need not adhere to that implied by the other parameters. The type given is set as a substring in the output file names for this exposure. This defaults to tst, for test.

force<bool> - Flag indicating that issues with the state of the system should be ignored, and the exposure executed anyway. Defaults to false.

x_binning<int> - Factor by which the x dimmension will be binned.

y_binning<int> - Factor by which the y dimmension will be binned.

seconds<double> - Target integration time for the exposure.

subordinate<bool> - Indicates that a peer system will actuate the shutter.

dark<bool> - Indicates that the shutter should remain disabled for this exposaure.

```
#include <camra_handler_doc.h>
```

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

- camra/camra_handler_doc.h

9.234 Exposure Class Reference

9.234.1 Detailed Description

Class that manages the state of an exposure and the capture of the corresponding data.

`#include <exposure.h>`

Public Member Functions

- **Exposure** (const ExposureDescriptor &d, CAMRAHardware &h, Shutter &shutter, PFIPClient &pfip, TCSClient &tcs, WeatherClient &weather, Sender *data_handler)
- void **execute** (void)
- void **abort** (bool write_data=true)
- void **pause** (void)
- bool **active** (void)
- ExposureDescriptor **get_descriptor** (void)
- void **wait_for** (camra_transition_t transition, int exposure, int observation)

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

- camra/exposure.h
- camra/exposure.cpp

9.235 ExposureDescriptor Struct Reference

9.235.1 Detailed Description

Static structure capturing information associated with the request for exposure.

`#include <exposure.h>`

Public Member Functions

- **ExposureDescriptor** (const CAMRAConfig &c)

Public Attributes

- bool debug_

  Debugging flag.

- string hostname_

  Hostname of the data acquisition computer.

- string instrument_

  Instrument in use.

- string object_

  Name of the object being observed.

- int observation_

  Observation identifier.

- int exposure_

  Sub-exposure ID, i.e. dither counter, etc.

- string recipe_

  Recipe used for this observation.

- string observatory_

  Observatory name.

- string observer_

  Name of observer.

- string operator_

  Name of operator.

- double duration_

  Requested exposure time.

- string type_

  Exposure type.

- string output_

  Initialized to output root and set to full exposure output path upon execution.

- string dirdate_

  Subdirectory date string, default is YYYYmmdd for current UT date.

- string storage_

  Location where data handler tool duplicates data.

- int x_pixels_

  Number of pixels in the X dimension.

- int y_pixels_

  Number of pixels in the Y dimension.

- int x_binning_

  Factor by which columns are to be binned in hardware.

- int y_binning_

  Factor by which rows are to be binned in hardware.

- bool force_flag_

  Force the exposure, ignoring any invalid states.

- bool subordinate_

  Read shutter info from the PFIP system even though the shutter is disabled.

- bool dark_

  Disable the shutter for this exposure and track EXPTIME locally.

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

- camra/exposure.h

9.236 ExposureInfo Struct Reference

Classes

- struct BINNING
- struct FILTER_INFO
- struct Image
- struct PlateScale
- struct ROI
- struct Sensor

Public Attributes

- struct {
  PixelFormat_t **next**
  PixelFormat_t **current**
  bool **changed**
  } **pixelFormat**

- struct {
  Image_t **next**
  Image_t **current**
  bool **changed**
  } **imageType**

- struct {
  BINNING **next**
  BINNING **current**
  bool **changed**
  } **binning**

- struct {
  ROI **next**
  ROI **current**
  bool **changed**
  } **roi**

- struct {
  double **next**
  double **current**
  bool **changed**
  } **exposureTime**

- struct {
  int **next**
  int **current**
  bool **changed**
  } **overScan**

- struct {
  FILTER_INFO **next**
  FILTER_INFO **current**

bool **changed**
int **currentFilterIndex**
int **sequenceIndex**
} **filterInfo**

- string **hostName**
- string **observatory**
- string **objectID**
- string **imageSource**
- Sensor **sensor**
- PlateScale **plateScale**
- Image **image**
- int **flushes**
- double **exposureOverhead**

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

- pas/include/Camera.h

9.237 tthread::fast_mutex Class Reference

9.237.1 Detailed Description

Fast mutex class. This is a mutual exclusion object for synchronizing access to shared memory areas for several threads. It is similar to the tthread::mutex class, but instead of using system level functions, it is implemented as an atomic spin lock with very low CPU overhead.

The `fast_mutex` class is NOT compatible with the `condition_variable` class (however, it IS compatible with the `lock_guard` class). It should also be noted that the `fast_mutex` class typically does not provide as accurate thread scheduling as a the standard `mutex` class does.

Because of the limitations of the class, it should only be used in situations where the mutex needs to be locked/unlocked very frequently.

Note:

The "fast" version of this class relies on inline assembler language, which is currently only supported for 32/64-bit Intel x86/AMD64 and PowerPC architectures on a limited number of compilers (GNU g++ and MS Visual C++). For other architectures/compilers, system functions are used instead.

`#include <fast_mutex.h>`

Public Member Functions

- fast_mutex ()
  Constructor.
- ~fast_mutex ()
  Destructor.
- void lock ()
- bool try_lock ()
- void unlock ()

9.237.2 Member Function Documentation

9.237.2.1 void tthread::fast_mutex::lock () [inline]

Lock the mutex. The method will block the calling thread until a lock on the mutex can be obtained. The mutex remains locked until `unlock()` is called.

See also:

lock_guard

9.237.2.2 bool tthread::fast_mutex::try_lock () [inline]

Try to lock the mutex. The method will try to lock the mutex. If it fails, the function will return immediately (non-blocking).

Returns:

`true` if the lock was acquired, or `false` if the lock could not be acquired.

9.237.2.3 void tthread::fast_mutex::unlock () [inline]

Unlock the mutex. If any threads are waiting for the lock on this mutex, one of them will be unblocked.

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

- het_lib/tthread/include/fast_mutex.h

9.238 FCU_Lamp Class Reference

The FCU_Lamp class manages the state of the lamps. There are 5 lamps on each FCU slider. The FCU_-Lamp class is also used for the single LDLS on the FCU.

`#include <FCUSubSystem.h>`

Public Member Functions

- pfip::LampState **GetInternalState** ()
- void **SetInternalState** (const pfip::LampState s)
- void **SetEnableTime** (const fptime t)
- fptime **GetEnableTime** ()
- void **SetWarmTime** (const fptime w)
- const fptime **GetWarmTime** ()

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

- pfip/include/FCUSubSystem.h

9.239 FCU_LDLS Class Reference

The class representing the LDLS and its state.

`#include <FCUSubSystem.h>`

Public Member Functions

- virtual FCU_Lamp & **GetLamp** (const int index=0)

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

- pfip/include/FCUSubSystem.h

9.240 FCU_Shutter Class Reference

Public Member Functions

- void **SetName** (string name)
- string **GetName** (void)
- virtual FCU_Lamp & **GetLamp** (const int index=0)=0
- virtual pfip::FCUShutterState **GetShutterStateActual** ()
- virtual pfip::FCUShutterState **GetShutterState** ()
- virtual void **SetShutterStateActual** (const pfip::FCUShutterState state)
- virtual void **SetShutterState** (const pfip::FCUShutterState state)
- virtual void **SetExposureTimeRemaining** (double millis)
- virtual double **GetExposureTimeRemaining** (void)

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

- pfip/include/FCUSubSystem.h

9.241 FCU_Slider Class Reference

A class to represent the FCU sliders. There are two -- slider a and slider b. Each contains five positions and five lamps.

`#include <FCUSubSystem.h>`

Public Member Functions

- pfip::SliderPos **GetPosActual** ()
- void **SetPosActual** (const pfip::SliderPos pos)
- pfip::SliderPos **GetPosRequested** ()
- void **SetPosRequested** (const pfip::SliderPos s)
- void **SetLampOn** (const int pos)
- void **SetLampOff** (const int pos)
- virtual FCU_Lamp & **GetLamp** (const int i=0)

Static Public Member Functions

- static pfip::SliderPos **ExtToIntPos** (int extPos)
- static int **IntToExtPos** (const pfip::SliderPos intPos)

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

- pfip/include/FCUSubSystem.h
- pfip/src/FCUSubsystem.cpp

9.242 FCU_State Class Reference

A class to hold state information for the FCU. Has status of FCU Head position and lightGuide position.

`#include <FCUSubSystem.h>`

Public Member Functions

- FCU_State ()
  Constructor.
- pfip::FCUState GetInternalState ()
  Return the state of the FCU.
- void SetInternalState (const pfip::FCUState s)
  Store the state of the FCU.
- std::string GetSliderName (const pfip::SelectorID pos)
  Return the current slider name.
- bool **GetSliderAJobComplete** ()
- bool **GetSliderBJobComplete** ()
- pfip::LightGuideSelectorPos GetLightGuideActual ()
  Return the current lightGuide position.
- pfip::LightGuideSelectorPos **GetLightGuideRequested** ()
- pfip::LightGuideSelectorPos **GetLightGuideAtFCUActual** ()
- pfip::LightGuideSelectorPos **GetLightGuideAtFCUHeadActual** ()
- bool **GetLightGuideJobComplete** (void)
- std::string GetLightGuideName (const pfip::LightGuideSelectorPos pos)
  Return the current lightGuide name.
- pfip::LightBankSelectorPos GetLightBankActual ()
  Return the current inputBank position.
- std::string GetLightBankName (const pfip::LightBankSelectorPos pos)
  Return the current inputBank position.
- pfip::LightBankSelectorPos GetLightBankRequested ()
  Return the last requested lightGuide position.
- bool **GetLightBankJobComplete** ()
- pfip::FCUHeadPos GetFCUHeadActual ()
  Return the current position of the FCU Head.
- bool **GetFCUHeadJobComplete** ()
- pfip::FCUHeadPos GetFCUHeadRequested ()
  Return the last requested FCU Head position.
- void SetFCUHeadRequested (const pfip::FCUHeadPos pos)

Set the requested FCU Head position.

- void SetFCUHeadActual (const pfip::FCUHeadPos pos)

  Set the current FCU Head position.

- void SetLightGuideAtFCUActual (pfip::LightGuideSelectorPos pos)

  Set the current lightGuide position.

- void **SetLightGuideRequested** (const pfip::LightGuideSelectorPos pos)
- void **SetLightGuideAtFCUHeadActual** (pfip::LightGuideSelectorPos pos)
- void **SetLightGuideJobInProgress** (void)
- void **SetLightGuideAtFCUJobComplete** (void)
- void **SetLightGuideAtFCUHeadJobComplete** (void)
- void SetLightBankJobInProgress (void)

  Set the current inputBank position.

- void **SetLightBankActual** (pfip::LightBankSelectorPos pos)
- void SetLightBankRequested (const pfip::LightBankSelectorPos pos)

  Change the requested inputBankGuide position.

- void **SetSliderAJobInProgress** (void)
- void **SetSliderBJobInProgress** (void)
- void **SetSliderAJobComplete** ()
- void **SetSliderBJobComplete** ()
- void **SetInputBankJobComplete** ()
- void **SetFCUHeadJobInProgress** ()
- void **SetFCUHeadJobComplete** ()

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

- pfip/include/FCUSubSystem.h
- pfip/src/FCUSubsystem.cpp

9.243 FCUStatus::fcu_status Struct Reference

Public Member Functions

- **BOOL_VAR** (lamp1a)
- **BOOL_VAR** (lamp1aAlert)
- **BOOL_VAR** (lamp2a)
- **BOOL_VAR** (lamp2aAlert)
- **BOOL_VAR** (lamp3a)
- **BOOL_VAR** (lamp3aAlert)
- **BOOL_VAR** (lamp4a)
- **BOOL_VAR** (lamp4aAlert)
- **BOOL_VAR** (lamp5a)
- **BOOL_VAR** (lamp5aAlert)
- **BOOL_VAR** (lamp1b)
- **BOOL_VAR** (lamp1bAlert)
- **BOOL_VAR** (lamp2b)
- **BOOL_VAR** (lamp2bAlert)
- **BOOL_VAR** (lamp3b)
- **BOOL_VAR** (lamp3bAlert)
- **BOOL_VAR** (lamp4b)
- **BOOL_VAR** (lamp4bAlert)
- **BOOL_VAR** (lamp5b)
- **BOOL_VAR** (lamp5bAlert)
- **BOOL_VAR** (ldls)
- **BOOL_VAR** (ldlsAlert)
- **DOUBLE_VAR** (lamp1atime, 6, 2)
- **DOUBLE_VAR** (lamp1awarm, 6, 2)
- **DOUBLE_VAR** (lamp2atime, 6, 2)
- **DOUBLE_VAR** (lamp2awarm, 6, 2)
- **DOUBLE_VAR** (lamp3atime, 6, 2)
- **DOUBLE_VAR** (lamp3awarm, 6, 2)
- **DOUBLE_VAR** (lamp4atime, 6, 2)
- **DOUBLE_VAR** (lamp4awarm, 6, 2)
- **DOUBLE_VAR** (lamp5atime, 6, 2)
- **DOUBLE_VAR** (lamp5awarm, 6, 2)
- **DOUBLE_VAR** (lamp1btime, 6, 2)
- **DOUBLE_VAR** (lamp1bwarm, 6, 2)
- **DOUBLE_VAR** (lamp2btime, 6, 2)
- **DOUBLE_VAR** (lamp2bwarm, 6, 2)
- **DOUBLE_VAR** (lamp3btime, 6, 2)
- **DOUBLE_VAR** (lamp3bwarm, 6, 2)
- **DOUBLE_VAR** (lamp4btime, 6, 2)
- **DOUBLE_VAR** (lamp4bwarm, 6, 2)
- **DOUBLE_VAR** (lamp5btime, 6, 2)
- **DOUBLE_VAR** (lamp5bwarm, 6, 2)
- **DOUBLE_VAR** (ldlstime, 6, 2)
- **DOUBLE_VAR** (ldlswarm, 6, 2)
- **INT_VAR** (sliderapos, 3)
- **INT_VAR** (sliderbpos, 3)

- **INT_VAR** (inputbankpos, 3)
- **INT_VAR** (lightguidepos, 3)
- QString **fcuheadpos** (void)
- **FCU_OPEN_CLOSED_VAR** (shutterastate)
- **FCU_OPEN_CLOSED_VAR** (shutterbstate)
- **FCU_OPEN_CLOSED_VAR** (ldlsshutterstate)
- **DOUBLE_VAR** (shutteraexposuretimeremaining, 6, 2)
- **DOUBLE_VAR** (shutterbexposuretimeremaining, 6, 2)
- **DOUBLE_VAR** (ldlsexposuretimeremaining, 6, 2)
- **STRING_VAR** (sliderALampNames)
- **STRING_VAR** (sliderBLampNames)
- **STRING_VAR** (sliderALampName)
- **STRING_VAR** (sliderBLampName)
- **STRING_VAR** (inputLightBankName)
- **STRING_VAR** (inputLightBankNames)
- **STRING_VAR** (outputLightGuideName)
- **STRING_VAR** (outputLightGuideNames)

Public Attributes

- int **m_fcuheadpos**

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

- gui/include/pfipStatusData.h

9.244 FCU_SubSystem Class Reference

The FCU_SubSystem class is used to control the operation of the Facility Calibration Unit hardware, and provides an API to pfipServer for its management. The FCU_SubSystem contains an instance of the MODBUSComm class, which provides communication to the hardware using the MODBUS protocol. In addition to providing operational changes to the FCU, FCU_SubSystem also maintains status of all the hardware and monitors alarms and notifications about changes to the hardware and its environment. The FCU_SubSystem uses a background thread to update these status changes.

`#include <FCUSubSystem.h>`

Classes

- struct **exposureInfo**

Public Member Functions

- FCU_SubSystem ()

  Constructor/Destructor for the FCU_SubSystem object.

- void **StartStatusThread** (void)
- void **StopStatusThread** (void)
- bool Initialize (const pfip::SelectorID selectorID, const fptime lampwarmsecs[5], const fptime ldlswarmsecs)

  Initialize state of FCU with lamp warmup times and LDLS configuration information,.

- bool **InitializeLamp** (const pfip::SelectorID selectorID, const std::string &lampName, const fptime warmtime)
- bool **InitializeLDLS** (const fptime lampTime)
- bool ShutDownFCU ()

  Shut down the FCU, turn off lamps, ldls, retract, but don't terminate the MODBUS connection.

- void CloseComm ()

  End the MODBUS connection and the notification thread.

- bool EnableLamps (pfip::SelectorID lampId, const pfip::LampState lamps[5], const pfip::LampState ldls)
- bool **EnableLDLS** ()
- bool **EnableLamps** (const pfip::SelectorID slider, const std::string &lampNameList, const pfip::LampState status)
- bool MoveSelectorPosition (pfip::SelectorID selectorID, pfip::SliderPos pos, bool wait)
- bool **MoveSelectorPosition** (pfip::SelectorID selectorID, std::string &lampName, bool wait)
- bool MoveOutputLightGuide (pfip::LightGuideSelectorPos band, bool wait)
- void **MoveFCUHead** (pfip::FCUHeadPos reqPosition, bool wait)
- bool ExposureSetup (pfip::SelectorID selectorID, pfip::SliderPos sliderPosition, const std::string &lampName, pfip::LightGuideSelectorPos lightGuideSelectorPos, const fptime &exposureTimeSecs, const bool fcuHeadOverride)
- bool **LDLSExposureSetup** (pfip::LightGuideSelectorPos lightGuidePos, const fptime &exposureTimeSecs, const bool fcuHeadOverride)
- void **LDLSExposureCleanup** ()

- void **ExposureCleanup** (const pfip::SelectorID selectorID)
- void **SaveCurrentExposureInfo** (pfip::SelectorID sliderID, pfip::SliderPos sliderPos, string lampName, pfip::LightBankSelectorPos lightBankSelectorPos, pfip::LightGuideSelectorPos lightGuideSelectorPos, double FCUExposureSecs, ACQ_SubSystem *pACQ)
- void **SaveLastExposureInfo** (pfip::SelectorID sliderID, pfip::SliderPos sliderPos, string lampName, pfip::LightBankSelectorPos lightBankSelectorPos, pfip::LightGuideSelectorPos lightGuideSelectorPos, double FCUExposureSecs, ACQ_SubSystem *pACQ)
- void **CancelExposure** (pfip::LightBankSelectorPos selectorPos)
- bool **WaitExposureComplete** (pfip::SelectorID sliderID, double timeout)
- int SetValue (int regNum, unsigned short *const val)
- int **GetValue** (int regNum, unsigned short *const val)
- Message * ReplyCurrentExposure ()
- Message * ReplyLastExposure ()
- void BroadcastStatusEvents ()

9.244.1 Member Function Documentation

9.244.1.1 void FCU_SubSystem::BroadcastStatusEvents ()

Send events with status updates

9.244.1.2 bool FCU_SubSystem::EnableLamps (pfip::SelectorID *lampId*, const pfip::LampState *lamps*[5], const pfip::LampState *ldls*)

Enable/disable lamps and/or ldls

9.244.1.3 bool FCU_SubSystem::ExposureSetup (pfip::SelectorID *selectorID*, pfip::SliderPos *sliderPosition*, const std::string & *lampName*, pfip::LightGuideSelectorPos *lightGuideSelectorPos*, const fptime & *exposureTimeSecs*, const bool *fcuHeadOverride*)

Setup slider position and exposure time for an FCU exposure

9.244.1.4 bool FCU_SubSystem::MoveOutputLightGuide (pfip::LightGuideSelectorPos *band*, bool *wait*)

Request the lightGuide head band to change. The method does not wait until the move has completed.

9.244.1.5 bool FCU_SubSystem::MoveSelectorPosition (pfip::SelectorID *selectorID*, pfip::SliderPos *pos*, bool *wait*)

Request a move of one of the sliders to the identified position. The method does not wait until the move has completed.

9.244.1.6 Message * FCU_SubSystem::ReplyCurrentExposure ()

Reply with state of the current exposure

9.244.1.7 Message * FCU_SubSystem::ReplyLastExposure ()

Reply with state of the last exposure

9.244.1.8 int FCU_SubSystem::SetValue (int *regNum*, unsigned short *const *val*)

Get and set individual MODBUS register

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

- pfip/include/FCUSubSystem.h
- pfip/src/FCUSubsystem.cpp

9.245 FCUCancelExposure Class Reference

9.245.1 Detailed Description

Cancel FCU exposure.

Required parameters:

shutter<string> - Shutter name.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUCancelExposure** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.246 FCUCommException Class Reference

Public Member Functions

- **FCUCommException** (int err)

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

- pfip/include/FCUCommunication.h

9.247 FCUCommunication Class Reference

Public Member Functions

- int **Connect** ()
- int **getValue** (int ValueID, unsigned short ∗Value)
- void **setValues** (int ValueID, int numValues, unsigned short ∗Values)
- void **close** ()
- void **sendCommand** ()
- void **lockEPICSAccess** ()
- void **initializeNotifications** ()

Static Public Member Functions

- static FCUCommunication ∗ **GetInstance** ()

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

- pfip/include/FCUCommunication.h

9.248 Ui::fcuControlTab Class Reference

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

- gui/build/ui_fcuControlTab.h

9.249 fcuControlTabUi Class Reference

Signals

- void **SendPFIPCommand** (const std::string, Message ∗)

Public Member Functions

- **fcuControlTabUi** (const char ∗name, QWidget ∗pLoadedWidget, guiConfig ∗pConfig)

Protected Attributes

- QList< string > **m_sliderALampNamesList**
- QList< string > **m_sliderBLampNamesList**
- QString **m_sliderALampNames**
- QString **m_sliderBLampNames**
- QString **m_inputLightBankNames**
- QString **m_outputLightGuideNames**

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

- gui/include/fcuControlTabUi.h
- gui/build/moc_fcuControlTabUi.cpp
- gui/src/fcuControlTabUi.cpp

9.250 FCUDeployHead Class Reference

9.250.1 Detailed Description

Deploy the Facility Control Unit (FCU) head for an exposure using the FCU shutter.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUDeployHead** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.251 FCUDisableAllLights Class Reference

9.251.1 Detailed Description

Turn off all the lamps and the ldls on the FCU.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUDisableAllLights** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.252 FCUEnableLamps Class Reference

9.252.1 Detailed Description

Toggle on or off specified FCU lamps.

Required parameters:

slider<string> - Identify the slider for the light source, either 'a' or 'b'.

lamps<string> - A string containing a comma-separated list of lamp names.

state<string> - \"on\" turns specified lamps on. \"off\" turns them off.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUEnableLamps** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.253 FCUEnableLightSources Class Reference

9.253.1 Detailed Description

Toggle on or off specified FCU lamps or Laser-Driven Light Source (LDLS).

Required parameters:

slider<string> - Identify the slider for the light source, either 'a' or 'b'.

Optional parameters:

lamp1<string> - lamp1=\"on\" turns FCU Lamp 1 on. lamp1=\"off\" turns FCU Lamp 1 off.

lamp2<string> - lamp2=\"on\" turns FCU Lamp 2 on. lamp2=\"off\" turns FCU Lamp 2 off.

lamp3<string> - lamp3=\"on\" turns FCU Lamp 3 on. lamp3=\"off\" turns FCU Lamp 3 off.

lamp4<string> - lamp4=\"on\" turns FCU Lamp 4 on. lamp4=\"off\" turns FCU Lamp 4 off.

lamp5<string> - lamp5=\"on\" turns FCU Lamp 5 on. lamp5=\"off\" turns FCU Lamp 5 off.

ldls<string> - ldls=\"on\" turns on the LDLS. ldls=\"off\" turns off the LDLS.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUEnableLightSources** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.254 FCUException Class Reference

Public Member Functions

- **FCUException** (int err)
- FCUException & **operator=** (const FCUException &)
- const int **what** ()

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

- pfip/include/FCUCommunication.h

9.255 FCUExposeLDLS Class Reference

9.255.1 Detailed Description

Take an exposure with the Laser-Driven Light Source

Required parameters:

time<float> - Number of seconds to expose.

Optional parameters:

headpos<string> - When this parameter is set to \"override\", do not require deployment of the FCU Head for the exposure. This option is used when the shutter is controlled by Virus. The default is to require deployment of the FCU Head for an exposure.

light_guide<string> - If the light_guide parameter is set to red (light_guide=\"red\"), use the light_guide's red setting. If the parameter is set to blue (light_guide=\"blue\"), use the blue setting. The default is to use the current light_guide setting for the exposure.

shutter<string> - If the shutter parameters is set to \"Virus\", then Virus will control the shutter. The command will return immediately after positioning the slider and triggering the exposure. If no shutter parameter is supplied, the focal plane shutter will be used to control the exposure. In this case, the command returns when the exposure is complete.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUExposeLDLS** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.256 FCUExposeSliderA Class Reference

9.256.1 Detailed Description

Take an exposure on FCU slider A.

Required parameters:

time<float> - Number of seconds to expose.

Optional parameters:

pos<int> - Slider position (lamp) to use for the exposure. An integer 1-5. Default is the current slider A position.

lamp_name<string> - The name of the lamp (string) to use for the exposure. Default is the current slider A position.

headpos<string> - When this parameter is set to \"override\", do not require deployment of the FCU Head for the exposure. This option is used when the shutter is controlled by Virus. The default is to require deployment of the FCU Head for an exposure.

light_guide<string> - If the light_guide parameter is set to red (light_guide=\"red\"), use the light_guide's red setting. If the parameter is set to blue (light_guide=\"blue\"), use the blue setting. The default is to use the current light_guide setting for the exposure.

shutter<string> - If the shutter parameters is set to \"Virus\", then Virus will control the shutter. The command will return immediately after positioning the slider and triggering the exposure. If no shutter parameter is supplied, the focal plane shutter will be used to control the exposure. In this case, the command returns when the exposure is complete.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUExposeSliderA** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.257 FCUExposeSliderB Class Reference

9.257.1 Detailed Description

Take an exposure on FCU slider B.

Required parameters:

time<float> - Number of seconds to expose.

Optional parameters:

pos<int> - Slider position (lamp) to use for the exposure. An integer 1-5. Default is the current slider B position.

lamp_name<string> - The name of the lamp (string) to use for the exposure. Default is the current slider B position.

headpos<string> - When this parameter is set to \"override\", do not require deployment of the FCU Head for the exposure. This option is used when the shutter is controlled by Virus. The default is to require deployment of the FCU Head for an exposure.

light_guide<string> - If the light_guide parameter is set to red (light_guide=\"red\"), use the light_guide's red setting. If the parameter is set to blue (light_guide=\"blue\"), use the blue setting. The default is to use the current light_guide setting for the exposure.

shutter<string> - If the shutter parameters is set to \"virus\", then Virus will control the shutter. The command will return immediately after positioning the slider and triggering the exposure. If no shutter parameter is supplied, the focal plane shutter will be used to control the exposure. In this case, the command returns when the exposure is complete. Default is the current slider B position.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUExposeSliderB** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.258 FCUGetCurrentExposureInfo Class Reference

9.258.1 Detailed Description

Reply with information about the current exposure.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUGetCurrentExposureInfo** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.259 FCUGetLastExposureInfo Class Reference

9.259.1 Detailed Description

Reply with information about the last exposure.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUGetLastExposureInfo** (Receiver ∗rcv, ACQ_SubSystem ∗acq, FCU_SubSystem ∗fcu)
- virtual void HandlerAction (Message ∗pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message ∗pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.260 FCUHandler Class Reference

Public Member Functions

- FCUHandler (const string &name, Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu, MessageHandlerDescriptor &usage)

  Constructor.

- pfip::SelectorID parseSlider (Message *pMessage)

  Common method to parse the slider id (A or B) parameter from command.

- void FCUExposeSlider (FCU_SubSystem *myFCU, pfip::SelectorID selectorID, Message *pMessage)

  Common method to perform an FCU exposure.

- bool parseSliderPos (pfip::SliderPos &pos, Message *pMessage)

  Common method to parse the slider position parameter from command.

- pfip::LightGuideSelectorPos **parseLightGuide** (Message *pMessage, bool bRequired)
- pfip::LightBankSelectorPos **parseLightBank** (Message *pMessage)

Protected Attributes

- FCU_SubSystem * **m_pFCU**

9.260.1 Constructor & Destructor Documentation

9.260.1.1 FCUHandler::FCUHandler (const string & *name*, Receiver * *rcv*, ACQ_SubSystem * *acq*, FCU_SubSystem * *fcu*, MessageHandlerDescriptor & *usage*) `[inline]`

Parameters:

← ***name*** - a std::string containing the name of the command handler. This is the name that will be used to identify the command

← ***rcv*** - a pointer to the receiver object associated with this handler

← ***acq*** - a pointer to the ACQ_SubSystem object that manages the PFIP acquisition system

← ***fcu*** - a pointer to the FCU_SubSystem object that manages the FCU subsystem

← ***usage*** - a std::string that describes the command for the common help()

9.260.2 Member Function Documentation

9.260.2.1 void FCUHandler::FCUExposeSlider (FCU_SubSystem * *myFCU*, pfip::SelectorID *selectorID*, Message * *pMessage*)

Parameters:

← **-** ptr to the FCU subsystem

← **-** sliderVal identifying the slider a or b

← **-** ptr to the Event object for the command

9.260.2.2 SelectorID FCUHandler::parseSlider (Message * *pMessage*)

Parameters:

→ - a reference to the output value identifying either slider a or b

← - a pointer to the event object associated with the command

9.260.2.3 bool FCUHandler::parseSliderPos (pfip::SliderPos & *pos*, Message * *pMessage*)

Parameters:

→ - reference to the output value identifying the position [1-5]

← - ptr to the Event object for the command

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.261 FCUInitLightSources Class Reference

9.261.1 Detailed Description

Configure warm-up times for the lamps and ldls in the FCU.

Required parameters:

slider<string> - Identify the slider for the light source, either 'a' or 'b'.

Optional parameters:

lamp1<float> - Number of seconds of warmup time necessary for Lamp 1.

lamp2<float> - Number of seconds of warmup time necessary for Lamp 2.

lamp3<float> - Number of seconds of warmup time necessary for Lamp 3.

lamp4<float> - Number of seconds of warmup time necessary for Lamp 4.

lamp5<float> - Number of seconds of warmup time necessary for Lamp 5.

ldls<float> - Number of seconds of warmup time necessary for LDLS.

```
#include <pfip_handler_doc.h>
```

Public Member Functions

- **FCUInitLightSources** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.262 FCUMoveLightBank Class Reference

9.262.1 Detailed Description

Change the input light bank setting.

Required parameters:

light_bank<string> - This parameter is set to either \"Slider A\", \"LDLS\", or \"Slider B\", depending on the light source desired.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUMoveLightBank** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.263 FCUMoveLightGuide Class Reference

9.263.1 Detailed Description

Change the output light guide setting.

Required parameters:

light_guide<string> - This parameter is set to either \"Red\" or \"Blue\", depending on the type of light desired.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUMoveLightGuide** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.264 FCUMoveSlider Class Reference

9.264.1 Detailed Description

Move to a different lamp on the specified slider.

Required parameters:

slider<string> - Specify the slider (e.g., \"a\" or \"b\")

Optional parameters:

pos<int> - Specify the lamp requested by position, 1-5. If lamp is requested by position, cannot use lamp_name parameter.

lamp_name<string> - Specify the lamp requested by configured name (e.g., \"Cd\"). This should be a name specified in the configuration for the slider requested. If lamp_name parameters is used to identify the lamp position, do not use the pos parameter.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUMoveSlider** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.265 FCUMutexException Class Reference

Public Member Functions

- **FCUMutexException** (int err)

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

- pfip/include/FCUCommunication.h

9.266 FCUPrintRegister Class Reference

9.266.1 Detailed Description

Display the value of an FCU modbus PLC register to the PFIP server log.

Required parameters:

regnum<int> - The register number to print.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUPrintRegister** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.267 FCURetractHead Class Reference

9.267.1 Detailed Description

Retract the Facility Control Unit (FCU) head.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCURetractHead** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.268 FCUSetLampWarmUp Class Reference

9.268.1 Detailed Description

Configure warm-up times for FCU lamps.

Required parameters:

slider<string> - Identify the slider for the light source, either 'a' or 'b'.

lamp_name<string> - The name of the lamp, as configured (e.g., \"Cd\" or \"Hg\"

warm_seconds<float> - Number of seconds of warmup time necessary for the lamp specified.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUSetLampWarmUp** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.269 FCUSetLDLSWarmUp Class Reference

9.269.1 Detailed Description

Configure warm-up time for laser-driven light source

Required parameters:

warm_seconds<float> - Number of seconds of warmup time.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUSetLDLSWarmUp** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.270 FCUShutDown Class Reference

9.270.1 Detailed Description

Shutdown the FCU.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUShutDown** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.271 FCUStatus Class Reference

Classes

- struct fcu_status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- fcu_status **m_fcuStatus**

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

- gui/include/pfipStatusData.h
- gui/src/pfipStatusData.cpp

9.272 Ui::fcuStatusTab Class Reference

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

- gui/build/ui_fcuStatusTab.h

9.273 fcuStatusTabUi Class Reference

Signals

- void **ForceHeartbeat** (std::string)

Public Member Functions

- **fcuStatusTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

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

- gui/include/fcuStatusTabUi.h
- gui/build/moc_fcuStatusTabUi.cpp
- gui/src/fcuStatusTabUi.cpp

9.274 FCUWriteRegister Class Reference

9.274.1 Detailed Description

Store an integer value into one of the FCU modbus registers.

Required parameters:

regnum<int> - The register number to write.

value<int> - The value to store in the register.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **FCUWriteRegister** (Receiver *rcv, ACQ_SubSystem *acq, FCU_SubSystem *fcu)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.275 fiducialDialog Class Reference

Public Member Functions

- **fiducialDialog** (const string &pipeline, QWidget ∗parent=0)

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

- gui/include/cameraImageScene.h

9.276 Ui::fiducialDialog Class Reference

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

- gui/build/ui_fiducialDlg.h

9.277 pasImagePipelineSubSystem::TCSData::fiducialObject Struct Reference

Public Attributes

- double **icX**
- double **icY**

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

- pas/include/pasImagePipelineSubSystem.h

9.278 ExposureInfo::FILTER_INFO Struct Reference

Public Attributes

- int **count**
- string **name** [FILTER_MAX_COUNT]
- int **sequence** [FILTER_MAX_COUNT]
- int **sequenceLength**
- double **exposureFactor** [FILTER_MAX_COUNT]
- double **zeroPoint** [FILTER_MAX_COUNT]
- double **magnitude** [FILTER_MAX_COUNT]

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

- pas/include/Camera.h

9.279 AcqCamStatus::filter_status Struct Reference

Public Member Functions

- **STRING_VAR** (currentPos)
- **INT_VAR** (currentIndex, 6)
- **STRING_VAR** (filterNames)

Public Attributes

- flags **m_flags**

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

- gui/include/pfipStatusData.h

9.280 Filters Class Reference

Public Member Functions

- **Filters** (TCSConfig *pConfig)

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

- pas/include/pasImagePipelineSubSystem.h

9.281 FitData Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual void **newData** (int size1, int size2)
- virtual void **deleteData** (void)

Public Attributes

- ushort **maxterm**
- double **pix2rad**
- double **pix2mm**
- double **wavelen**
- gsl_matrix * **Ufit**
- gsl_matrix * **Vfit**
- gsl_matrix * **Chol**
- gsl_matrix * **invChol**
- gsl_vector * **Sfit**
- gsl_vector * **Bfit**
- gsl_vector * **Xfit**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.282 tcs_photometry::FitPars Struct Reference

9.282.1 Detailed Description

result of PSF fit, bivariate Gaussian and bivariate Moffat

`#include <object.h>`

Public Attributes

- ltl::FVector< double, 7 > **gauss_par_**
- ltl::FVector< double, 7 > **e_gauss_par_**
- ltl::FVector< double, 8 > **moffat_par_**
- ltl::FVector< double, 8 > **e_moffat_par_**
- double **gauss_chi2_**
- double **moffat_chi2_**
- double **gauss_flux_**
- double **gauss_dflux_**
- double **moffat_flux_**
- double **moffat_dflux_**
- double **gauss_m_**
- double **gauss_dm_**
- double **moffat_m_**
- double **moffat_dm_**

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

- het_lib/photometry/include/object.h

9.283 pasImagePipelineSubSystem::FITSMapEntry Struct Reference

Public Attributes

- string **keyWord**
- string **comment**
- FITSItemType_t **type**

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

- pas/include/pasImagePipelineSubSystem.h

9.284 FitsOutWrapper Class Reference

Public Member Functions

- **FitsOutWrapper** (const std::string &filename)
- **FitsOutWrapper** (unsigned char **inmemptr, size_t *inmemlen)
- void **add_header_field** (const std::string &keyword, const bool value, const std::string &comment="")
- void **add_header_field** (const std::string &keyword, const int value, const std::string &comment="")
- void **add_header_field** (const std::string &keyword, const float value, const std::string &comment="")
- void **add_header_field** (const std::string &keyword, const double value, const std::string &comment="")
- void **add_header_field** (const std::string &keyword, const std::string &value, const std::string &comment="")
- void **add_header_field** (const std::string &keyword, const char *value, const std::string &comment="")

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

- het_lib/common/include/fits_helper.h

9.285 fixed_point::FixedPoint< T, N > Class Template Reference

9.285.1 Detailed Description

template<typename T, unsigned int N> class fixed_point::FixedPoint< T, N >

Fixed-precision datatype with

- built-in integer type T, and
- N decimal digits in fractional part

`#include <fixed.h>`

Public Types

- typedef T **raw_int_type**

Public Member Functions

- **FixedPoint** (const FixedPoint< T, N > &other)
- FixedPoint (const raw_int_type integer, const raw_int_type fraction)
- FixedPoint (const raw_int_type integer)
- FixedPoint< T, N > & operator= (const FixedPoint< T, N > &other)
  assignment of floating point value
- void set_raw_value (const raw_int_type raw_bits)
  set the internal raw value
- template<typename Char , class Traits >
  bool read (std::basic_istream< Char, Traits > &strm)
  read from input stream
- raw_int_type round () const
  round to nearest integer
- raw_int_type integer () const
  return integer part
- raw_int_type fraction () const
  return fractional part
- raw_int_type value () const
  return the raw underlying native value
- FixedPoint (const long double value)
  construct from floating point value
- **FixedPoint** (const double value)

- **FixedPoint** (const float value)
- FixedPoint< T, N > & operator= (const double value)

  assignment of floating point value

- FixedPoint< T, N > & **operator=** (const long double value)
- long double as_long_double () const

  type conversion

- **operator long double** () const
- double **as_double** () const
- **operator double** () const
- float **as_float** () const
- **operator float** () const
- std::string **as_string** () const
- **operator std::string** () const
- FixedPoint< T, N > & **operator+=** (const FixedPoint< T, N > f)
- FixedPoint< T, N > & **operator-=** (const FixedPoint< T, N > f)
- FixedPoint< T, N > & **operator*=** (const FixedPoint< T, N > f)
- FixedPoint< T, N > & **operator/=** (const FixedPoint< T, N > f)
- FixedPoint< T, N > & **operator++** ()
- FixedPoint< T, N > **operator++** (int)
- FixedPoint< T, N > & **operator--** ()
- FixedPoint< T, N > **operator--** (int)
- FixedPoint< T, N > **operator-** () const

Static Public Attributes

- static raw_int_type const places = N

  Number of decimal places.

- static raw_int_type const places10 = fixed_internal::power10<N>::value

  pow(10,N) used to scale fractional part

9.285.2 Constructor & Destructor Documentation

9.285.2.1 template<typename T, unsigned int N> fixed_point::FixedPoint< T, N >::FixedPoint (const raw_int_type *integer*, const raw_int_type *fraction*) [inline]

Construct from separate integer and fractional parts, padding the fraction with leading zeros, e.g., assuming `FixedPoint<int64,2>`, initialize to 123.45 with `Fixed(123, 45)`. Initialize to 12.07 with `Fixed(12, 7)`.

9.285.2.2 template<typename T, unsigned int N> fixed_point::FixedPoint< T, N >::FixedPoint (const raw_int_type *integer*) [inline]

Construct from integer Note that this sets only the integer part of the number. This is not initialization from a raw `raw_int_type` Use `set_raw_value` () for this purpose.

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

- het_lib/common/include/fixed.h

9.286 PFIPStatusBase::flags Struct Reference

Public Member Functions

- **BOOL_VAR** (powerStatus)
- **BOOL_VAR** (configured) BOOL_VAR(enabled)
- **BOOL_VAR** (reached)
- **BOOL_VAR** (fault)
- **BOOL_VAR** (warning)
- **BOOL_VAR** (setpoint)
- **BOOL_VAR** (following)
- **BOOL_VAR** (limit)
- **BOOL_VAR** (powerStatusAlert)
- **BOOL_VAR** (configuredAlert)
- **BOOL_VAR** (enabledAlert)
- **BOOL_VAR** (reachedAlert)
- **BOOL_VAR** (faultAlert)
- **BOOL_VAR** (warningAlert)
- **BOOL_VAR** (setpointAlert)
- **BOOL_VAR** (followingAlert)
- **BOOL_VAR** (limitAlert)

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

- gui/include/pfipStatusData.h

9.287 flagValues Class Reference

Public Attributes

- int **radius**
- double **kappa**
- float **nan_value**
- int **verbose**
- bool **quiet_please**

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

- het_lib/photometry/src/psf_zernickes.cpp

9.288 focalPlaneShutterScene Class Reference

Public Member Functions

- **focalPlaneShutterScene** (QGraphicsView *pView, focalPlaneShutterTabUi *pTab)
- void **moveShutter** (unsigned int encoderModulo)
- void **rotateMotor** ()

Protected Attributes

- QGraphicsView * **m_pView**
- focalPlaneShutterTabUi * **m_pTab**
- int **m_motorTimeouts**

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

- gui/include/focalPlaneShutter.h
- gui/src/focalPlaneShutter.cpp

9.289 Ui::focalPlaneShutterTab Class Reference

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

- gui/build/ui_focalPlaneShutterTab.h

9.290 focalPlaneShutterTabUi Class Reference

Signals

- void **SendPFIPCommand** (const std::string, Message ∗)
- void **ForceHeartbeat** (std::string)

Public Member Functions

- **focalPlaneShutterTabUi** (const char ∗name, QWidget ∗pLoadedWidget, guiConfig ∗pConfig)
- bool **eventFilter** (QObject ∗pObject, QEvent ∗pEvent)
- void **resize** (void)

Protected Attributes

- focalPlaneShutterScene ∗ **m_pShutterScene**
- QTimer **m_motorTimer**

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

- gui/include/focalPlaneShutterTabUi.h
- gui/build/moc_focalPlaneShutterTabUi.cpp
- gui/src/focalPlaneShutterTabUi.cpp

9.291 Foo Class Reference

Static Public Member Functions

- static void **jumper** (int sig)

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

- playground/try_catch/try_catch.cpp

9.292 FooBar Class Reference

Public Member Functions

- void **write_value** (void)
- void **read_value** (void)

Public Attributes

- tthread::thread **t1_**
- tthread::thread **t2_**
- int **value_**

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

- playground/thread_safety/test_thread_safety.cpp

9.293 force_state_event Class Reference

9.293.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **force_state_event** (MetrologyLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.293.2 Member Function Documentation

9.293.2.1 void force_state_event::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.294 ForceStatusEvent Class Reference

9.294.1 Detailed Description

Force a status event from the indicated pipeline The pipeline (camera) name, is prepended to the handler name (i.e. Guider1_).

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.295 ForceStatusEventsHandlerClass Class Reference

Public Member Functions

- **ForceStatusEventsHandlerClass** (TCSSubSystem *server)
- virtual void process (Message *pMessage)
- **ForceStatusEventsHandlerClass** (TCSSubSystem *server)
- virtual void process (Message *pMessage)

9.295.1 Member Function Documentation

9.295.1.1 virtual void ForceStatusEventsHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements ServerHandler.

9.295.1.2 virtual void ForceStatusEventsHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements ServerHandler.

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

- legacy/include/legacyServer.h
- pointing/include/pointingServer.h

9.296 formatChunk Class Reference

Public Attributes

- chunkHeader **header**
- unsigned short **wFormatTag**
- unsigned short **wChannels**
- unsigned int **dwSamplesPerSec**
- unsigned int **dwAverageBytesPerSec**
- unsigned short **wBlockAlign**
- unsigned short **dwBitsPerSample**

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

- gui/include/guiSound.h

9.297 fptime Class Reference

9.297.1 Detailed Description

Public Member Functions

- fptime ()
- fptime (const fptime &t)
- fptime (int64_t raw)
- fptime (const FixedTime &f)
- fptime (int s, int n)
- fptime & **operator=** (const fptime &rhs)
- fptime & **operator+=** (const fptime &f)
- fptime & **operator-=** (const fptime &f)

- fptime (index_time i)
- **fptime** (julian_day jd)
- **fptime** (modified_julian_day mjd)
- **fptime** (utc_iso8601 utc_string)
- **fptime** (utc u)
- utc as_utc () const
- utc_iso8601 **as_utc_iso8601** () const
- utc_fits **as_utc_fits** () const
- utc_date **as_utc_date** () const
- utc_time_of_day **as_utc_time_of_day** () const
- index_time **as_index_time** () const
- julian_day **as_jd** () const
- modified_julian_day **as_mjd** () const
- double **as_decimal_year** () const

Static Public Member Functions

- static fptime now (void)
- static fptime from_decimal_equinox (double equinox)

9.297.2 Constructor & Destructor Documentation

9.297.2.1 fptime::fptime ()

Construct a fixed precision time initialized to zero.

9.297.2.2 fptime::fptime (const fptime & *t*) `[inline]`

Copy constructor; implicit declaration of copy constructors is deprecated

9.297.2.3 fptime::fptime (int64_t *raw*)

Construct a fixed precision time from the given int64. This assumes that the int64 has been initialized with the raw value of an fptime object as acquired by fptime::value().

9.297.2.4 fptime::fptime (const FixedTime & *f*)

Construct from base class.

9.297.2.5 fptime::fptime (index_time *i*)

Construct a fixed precision time on the UTC calendar from the given time in one of the alternate supported calendars that are implicit in the method/parameter names below.

9.297.2.6 fptime::fptime (int *s*, int *n*)

Construct a fixed precision time from integer (s) and fractional (n) parts.

9.297.3 Member Function Documentation

9.297.3.1 utc fptime::as_utc () const

Conversion routines returning broken down, string, and alternate calendar representations.

9.297.3.2 fptime fptime::from_decimal_equinox (double *equinox*) [static]

Returns the fptime corresponding to the given equinox (in decimal year).

9.297.3.3 fptime fptime::now (void) [static]

Get the current system time.

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

- het_lib/common/include/fptime.h
- het_lib/common/src/fptime.cpp

9.298 fw102b::FW102B Class Reference

Public Member Functions

- def **__init__**
- def **connect**
- def **disconnect**
- def **command**
- def **set**
- def **status**
- def **version**

Public Attributes

- **name**
- **port**
- **host**
- **log**
- **csock**
- **rfile**
- **wfile**
- **lock**
- **connected**

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

- integration/fw102b.py

9.299 GenerateExceptionHandler Class Reference

Public Member Functions

- virtual void process (Message ∗message)

9.299.1 Member Function Documentation

9.299.1.1 virtual void GenerateExceptionHandler::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- het_lib/messaging/include/builtin_handlers.h

9.300 get Class Reference

Public Member Functions

- **get** (Config &d, const MessageHandlerDescriptor &m)
- void process (Message *message)
- **get** (IglooSensors &d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.300.1 Member Function Documentation

9.300.1.1 void get::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

9.300.1.2 void get::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- ics/config.h
- igloo_sensors/igloo_sensors.h

9.301 get_analysis_region Class Reference

9.301.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_analysis_region** (GuideLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.301.2 Member Function Documentation

9.301.2.1 void get_analysis_region::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.302 camra_handlers::get_ccd_temp Class Reference

Read the temperature of the given CCD and replies with the result.

9.302.1 Detailed Description

Parameters:

ccd<***string***> - identifier of the CCD to operate on, defaults to 0 for all.

update_state<***bool***> - flag indicating when true that the hardware should be queried, defaults to returning that last value obtained.

```
#include <camra_handlers.h>
```

Public Member Functions

- **get_ccd_temp** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.302.2 Member Function Documentation

9.302.2.1 void camra_handlers::get_ccd_temp::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.303 get_ccd_temp Class Reference

9.303.1 Detailed Description

Read the temperature of the given CCD. Temperatures are returned by HRL (XXX document HRL format) key followed by a \".temp\" prefix.

Optional parameters:

ccd<int> - Identifier of the CCD to operate on, defaults to 0 for all.

update_state<bool> - Flag indicating to read a new value from the hardware. Default behavior is to return the previous value read incurring no additional overhead in hardware latencies.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.304 camra_handlers::get_clock_voltages Class Reference

Public Member Functions

- **get_clock_voltages** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.304.1 Member Function Documentation

9.304.1.1 void camra_handlers::get_clock_voltages::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.305 get_clock_voltages Class Reference

9.305.1 Detailed Description

Returns the clock voltages for each CCD in the system.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.306 get_correction Class Reference

9.306.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_correction** (MetrologyLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.306.2 Member Function Documentation

9.306.2.1 void get_correction::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.307 camra_handlers::get_cryo_pressure Class Reference

Public Member Functions

- **get_cryo_pressure** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.307.1 Member Function Documentation

9.307.1.1 void camra_handlers::get_cryo_pressure::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.308 get_cryo_pressure Class Reference

9.308.1 Detailed Description

Read the cryostat pressure of the spectrograph for the given controller. Temperatures are returned by HRL (XXX document HRL format) key followed by a \".cryo_pressure\" prefix.

Optional parameters:

cntl<int> - Identifier of the controller to operate on, defaults to 0 for all.

update_state<bool> - Flag indicating to read a new value from the hardware. Default behavior is to return the previous value read incurring no additional overhead in hardware latencies.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.309 camra_handlers::get_cryo_temp Class Reference

Public Member Functions

- **get_cryo_temp** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.309.1 Member Function Documentation

9.309.1.1 void camra_handlers::get_cryo_temp::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.310 get_cryo_temp Class Reference

9.310.1 Detailed Description

Read the \"cold finger\" temperature of the given controller. Temperatures are returned by HRL (XXX document HRL format) key followed by a \".cryo_temp\" prefix.

Optional parameters:

cntl<int> - Identifier of the controller to operate on, defaults to 0 for all.

update_state<bool> - Flag indicating to read a new value from the hardware. Default behavior is to return the previous value read incurring no additional overhead in hardware latencies.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.311 get_current_local_sidereal_time Class Reference

9.311.1 Detailed Description

Returns the current LST, at the HET.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_current_local_sidereal_time** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.311.2 Member Function Documentation

9.311.2.1 void get_current_local_sidereal_time::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.312 get_current_pointing Class Reference

9.312.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_current_pointing** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.312.2 Member Function Documentation

9.312.2.1 void get_current_pointing::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.313 get_current_trajectory_info Class Reference

9.313.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_current_trajectory_info** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.313.2 Member Function Documentation

9.313.2.1 void get_current_trajectory_info::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.314 camra_handlers::get_exposure_info Class Reference

Public Member Functions

- **get_exposure_info** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.314.1 Member Function Documentation

9.314.1.1 void camra_handlers::get_exposure_info::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.315 get_exposure_info Class Reference

9.315.1 Detailed Description

Returns the exposure descriptor fields associated with the currently active exposure.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.316 get_fiducial Class Reference

9.316.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_fiducial** (GuideLoop *l, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.316.2 Member Function Documentation

9.316.2.1 void get_fiducial::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.317 get_focus_gain Class Reference

9.317.1 Detailed Description

Returns the current gain factor applied to the focus correction emitted by the loop.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_focus_gain** (WFSLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.317.2 Member Function Documentation

9.317.2.1 void get_focus_gain::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.318 get_focus_offset Class Reference

9.318.1 Detailed Description

Returns the current focus offset.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_focus_offset** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.318.2 Member Function Documentation

9.318.2.1 void get_focus_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.319 get_focus_window_length Class Reference

9.319.1 Detailed Description

Returns the number of samples that are averaged to produce a single focus correction from the WFS loop. The magnitude of the correction is the resulting average divided by the number of samples.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_focus_window_length** (WFSLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.319.2 Member Function Documentation

9.319.2.1 void get_focus_window_length::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.320 get_hardware_status Class Reference

9.320.1 Detailed Description

Read all hardware status fields for the given CCD.

Optional parameters:

update_state<bool> - Flag indicating to read a new value from the hardware. Default behavior is to return the previous value read incurring no additional overhead in hardware latencies.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.321 camra_handlers::get_hardware_status Class Reference

Public Member Functions

- **get_hardware_status** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.321.1 Member Function Documentation

9.321.1.1 void camra_handlers::get_hardware_status::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.322 get_heater_voltage Class Reference

9.322.1 Detailed Description

Read the CCD heater voltage. Voltages are returned by HRL (XXX document HRL format) key followed by a \".voltage\" prefix.

Optional parameters:

ccd<int> - Identifier of the CCD to operate on, defaults to 0 for all.

update_state<bool> - Flag indicating to read a new value from the hardware. Default behavior is to return the previous value read incurring no additional overhead in hardware latencies.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.323 camra_handlers::get_heater_voltage Class Reference

Public Member Functions

- **get_heater_voltage** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.323.1 Member Function Documentation

9.323.1.1 void camra_handlers::get_heater_voltage::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.324 get_itf_offset Class Reference

9.324.1 Detailed Description

Returns the current offset of the ITF sphere.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_itf_offset** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.324.2 Member Function Documentation

9.324.2.1 void get_itf_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.325 get_metadata Class Reference

9.325.1 Detailed Description

Returns all key/value pairs set through set_metadata.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_metadata** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.325.2 Member Function Documentation

9.325.2.1 void get_metadata::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.326 get_next_trajectory_info Class Reference

9.326.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_next_trajectory_info** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.326.2 Member Function Documentation

9.326.2.1 void get_next_trajectory_info::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.327 get_radec_offset Class Reference

9.327.1 Detailed Description

Returns the current offset in apparent RA and DEC in arcseconds relative to the coordinates stored in the radec offset register.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_radec_offset** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.327.2 Member Function Documentation

9.327.2.1 void get_radec_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.328 get_sky_offset Class Reference

9.328.1 Detailed Description

Returns the current angular offset on sky.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_sky_offset** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.328.2 Member Function Documentation

9.328.2.1 void get_sky_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.329 get_status Class Reference

9.329.1 Detailed Description

Returns the current status of the probe. Throws an exception if the status is unknown. XXX document info returned as it evolves.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_status** (TCSDriver *d, PFIPClient &pfip, Instrument &instrument, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.329.2 Member Function Documentation

9.329.2.1 void get_status::process (Message * *message*) [virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/pfip_client.h
- tcs/pfip_client.cpp

9.330 get_tiptilt_gain Class Reference

9.330.1 Detailed Description

Returns the current gain factor applied to the tiptilt correction emitted by the loop.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_tiptilt_gain** (WFSLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.330.2 Member Function Documentation

9.330.2.1 void get_tiptilt_gain::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.331 get_tiptilt_offset Class Reference

9.331.1 Detailed Description

Returns the current angular offset on tiptilt.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_tiptilt_offset** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.331.2 Member Function Documentation

9.331.2.1 void get_tiptilt_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.332 get_tiptilt_window_length Class Reference

9.332.1 Detailed Description

Returns the number of samples that are averaged to produce a single tiptilt correction from the WFS loop. The magnitude of the correction is the resulting average divided by the number of samples.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_tiptilt_window_length** (WFSLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.332.2 Member Function Documentation

9.332.2.1 void get_tiptilt_window_length::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.333 get_tuning< T > Class Template Reference

9.333.1 Detailed Description

template<typename T> class get_tuning< T >

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **get_tuning** (MetrologyLoop *l, MetrologyPIDLoop< T > *p, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.333.2 Member Function Documentation

9.333.2.1 template<typename T > void get_tuning< T >::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.334 GetACQSystemInfo Class Reference

9.334.1 Detailed Description

Return a reply message containing status and encoder positions for the focal plane shutter, the acquisition camera, ENWA filter, acquisition camera filter, and dither setting.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **GetACQSystemInfo** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.335 GetAnalysisRegion Class Reference

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

- pas/include/pas_handler_doc.h

9.336 GetExposureTime Class Reference

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

- pas/include/pas_handler_doc.h

9.337 GetGain Class Reference

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

- pas/include/pas_handler_doc.h

9.338 GetLightSourcePowerLevel Class Reference

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

- pas/include/pas_handler_doc.h

9.339 GetOverScan Class Reference

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

- pas/include/pas_handler_doc.h

9.340 GetPhotPhotometryOptions Class Reference

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

- pas/include/pas_handler_doc.h

9.341 GetPosition Class Reference

Get current dome position.

9.341.1 Detailed Description

Parameters:

none

`#include <legacyDomeSubSystem.h>`

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

- legacy/include/legacyDomeSubSystem.h

9.342 GetProbeInfo Class Reference

9.342.1 Detailed Description

Return a reply message containing information about probe and filter status.

Required parameters:

probe<string> - Identify the probe for which information is requested. One of: 'Guider1', 'Guider2', 'WFS1', or 'WFS2'.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **GetProbeInfo** (Receiver ∗rcv, ACQ_SubSystem ∗acq)
- virtual void HandlerAction (Message ∗pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message ∗pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.343 GetProbePosition Class Reference

9.343.1 Detailed Description

Return a reply message containing information about probe position and status.

Required parameters:

probe<string> - Identify the probe for which information is requested. One of: 'Guider1', 'Guider2', 'WFS1', or 'WFS2'.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **GetProbePosition** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.344 GetRegionOfInterest Class Reference

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

- pas/include/pas_handler_doc.h

9.345 GetTemperatures Class Reference

Get temperatures.

9.345.1 Detailed Description

Parameters:

none

`#include <pasImagePipelineSubSystem.h>`

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

- pas/include/pas_handler_doc.h

9.346 GetWeatherData Class Reference

Return legacy Weather data items.

9.346.1 Detailed Description

Parameters:

none

`#include <legacyWeatherSubSystem.h>`

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

- legacy/include/legacyWeatherSubSystem.h

9.347 GetWFSPhotometryOptions Class Reference

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

- pas/include/pas_handler_doc.h

9.348 globals Struct Reference

Public Attributes

- bool **help**
- bool **quiet**
- bool **guider_mode**
- bool **ttcam_mode**
- bool **residual_images_**
- bool **header_config_**
- int **fits_ext_**
- float **pixscale_**
- ConfigureParams **pp**
- GuideRect **gr**
- int **coord** [4]
- string **traj_type_s**
- double **ra**
- double **dec**
- string **dir**
- double **elevation**
- string **azrest**
- double **equinox**
- string **dbsource**
- bool **verbose**
- double **ra_offset**
- double **azimuth**
- string **ephemeris_file**
- string **sat_file**
- string **object**
- string **traj_file**
- bool **mount_model**
- string **ser7path**
- string **outfile**
- string **invfile**
- double **jd**
- double **xin**
- double **yin**
- string **pier_model**

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

- het_lib/photometry/src/testphot.cpp
- integration/sir/Het_trajectory/gettraj.cpp
- tcs/traj_comp/point_test.cpp

9.349 GLTrackerConfig Class Reference

Public Member Functions

- **GLTrackerConfig** (bool isNamed=0)

Public Attributes

- std::string **tcs_event_route**
- bool **label_mirrors**

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

- playground/gltracker/gltracker_config.h

9.350 GLTrackerDriver Class Reference

Classes

- struct tracker_position

Public Member Functions

- **GLTrackerDriver** (const string &appName, GLTrackerConfig *config)
- int **start** ()
- void **stop** ()
- void **reshape** (int x, int y)
- void **keyboard** (unsigned char key, int x, int y)
- void **special** (int key, int x, int y)
- void **display** (void)
- void **idle** (void)

Protected Member Functions

- void **init_opengl** (int argc, char *argv[ ])
- void **print_key_help** ()
- void **event_thread** ()
- void **drawCoordinateAxes** (double width)
- void **drawTrackerSphere** (double width)
- void **drawTracker** (const tracker_position &p)
- void drawHexagon3D (const vec3f &c, const vec3f &n, const float radius, const int j, const int k)
- void drawPrimaryMirror (void)
- void **drawHistory** (void)
- void drawTrajectory (void)

  object transformation

Protected Attributes

- double **scale**
- double **viewangle**
- double **tippangle**
- double **xtrans**
- double **ytrans**
- double ztrans

  tracker ITF coordinates

- tracker_position current_p

  window size

- int **win_width**
- int win_height

  event route to listen to point events

- std::string tcs_event_route
  history of tracker motion
- std::list< tracker_position > history
  current trajectory
- std::list< tracker_position > trajectory
  mutex to synchronize display routines
- tthread::fast_mutex display_mutex
  handle on the configuration object
- GLTrackerConfig * **config**

9.350.1 Member Function Documentation

9.350.1.1 void GLTrackerDriver::drawHexagon3D (const vec3f & *c*, const vec3f & *n*, const float *radius*, const int *j*, const int *k*) [protected]

Draw a hexagon in a plane defined by the surface normal vector `n` centered at `c` with radius `radius` (the radius is the distance from the center to a vertex.

9.350.1.2 void GLTrackerDriver::drawPrimaryMirror (void) [protected]

Draw the 91 segments of the primary mirror

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

- playground/gltracker/gltracker_driver.h
- playground/gltracker/gltracker_driver.cpp

9.351 go_next Class Reference

9.351.1 Detailed Description

Sets the system up for the next trajectory. The call returns on exception or when the tracker is on trajectory. If positioning of additional telescope components is requested, those are assumed to be settled and corrected for upon return.

Optional parameters:

move_dome<bool> - Flag indicating that the dome should be moved to the Azimuth of the requested trajectory. If given, and any error is encountered in positioning the dome, the trajectory will be aborted and the corresponding exception thrown.

move_structure<bool> - Flag indicating that the stucture should be moved to the Azimuth of the requested trajectory. If given, and any error is encountered in positioning the structure, the trajectory will be aborted and the corresponding exception thrown.

move_probes<bool> - Flag indicating that any configured PFIP metrology probes should be moved to the requested position.

See also:

set_position

override_structure_correction<bool> - Flag indicating that the correction of the trajectory, for the error in the placement of the structure, should be skipped.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **go_next** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.351.2 Member Function Documentation

9.351.2.1 void go_next::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.352 GoNextFlags Class Reference

9.352.1 Detailed Description

Class used to capture the runtime flags of the go_next() handler call.

`#include <tcs_driver.h>`

Public Member Functions

- **GoNextFlags** (Message &m)

Public Attributes

- const bool & **move_dome**
- const bool & **move_structure**
- const bool & **move_probes**
- const bool & **correct_for_structure**
- const bool & **override_structure_correction**
- const bool & **correct_parallactic**

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

- tcs/tcs_driver.h
- tcs/tcs_driver.cpp

9.353 GroupConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

- virtual void print_summary (void)

  Called prior to displaying options.

Public Attributes

- int **m_numOutlets**
- string **m_outlet** [20+1]
- string **m_color**

9.353.1 Member Function Documentation

9.353.1.1 virtual void GroupConfig::print_summary (void) [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.353.1.2 void GroupConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- apc/include/apcConfig.h
- apc/src/apcConfig.cpp

9.354 guiConfig Class Reference

Public Member Functions

- **guiConfig** (bool isNamed=false)
- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

- virtual void print_summary (void)

  Called prior to displaying options.

- **guiConfig** (bool isNamed=0)
- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

- virtual void print_summary (void)

  Called prior to displaying options.

Public Attributes

- bool **m_bValidateLoginIds**
- string **m_telescopeOperator** [50+1]
- string **m_loginId**
- string **m_mainWindowTabName** [50+1]
- string **m_statusEventSystems** [15+1]
- string **m_heartbeatEventSystems** [20+1]
- string **m_errorMonitorSystems** [15+1]
- string **m_errorMonitorSystemFilter**
- string **m_errorMonitorSourceFilter**
- string **m_errorMonitorKeyFilter**
- int **m_errorMonitorMaxEvents**
- string **m_errorMonitorDBPathname**
- string **m_eventMonitor1Systems** [15+1]
- string **m_eventMonitor1SystemFilter**
- string **m_eventMonitor1SourceFilter**
- string **m_eventMonitor1KeyFilter**
- int **m_eventMonitor1MaxEvents**
- string **m_eventMonitor1DBPathname**
- string **m_warningSoundFilename**
- string **m_errorSoundFilename**
- string **m_fatalErrorSoundFilename**
- string **m_alarmSoundFilename**
- int **m_LRS2BSpecId**
- string **m_videoSwitchAddress**

- struct {
  string **m_labels** [(16+3+1+5)+1]
  string **m_numbers** [(16+3+1+5)+1]
  string **m_descriptions** [(16+3+1+5)+1]
  string **m_commands** [(16+3+1+5)+1]
  string **m_preSelect**
  } **m_videoSwitch**

- double **m_virusMinTargetSetpoint**
- double **m_virusMaxTargetSetpoint**
- double **m_virusWarmUpTargetSetpoint**
- double **m_virusCoolDownTargetSetpoint**
- struct {
  string **m_labels** [(16+3+1+5)+1]
  string **m_numbers** [(16+3+1+5)+1]
  string **m_descriptions** [(16+3+1+5)+1]
  string **m_commands** [(16+3+1+5)+1]
  string **m_preSelect**
  } **m_videoSwitch**

9.354.1 Member Function Documentation

9.354.1.1 virtual void guiConfig::print_summary (void) [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.354.1.2 virtual void guiConfig::print_summary (void) [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.354.1.3 virtual void guiConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

9.354.1.4 void guiConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- gui/include/guiConfig.h
- utility_gui/include/guiConfig.h
- gui/src/guiConfig.cpp
- utility_gui/src/guiConfig.cpp

9.355 guideBoxDialog Class Reference

Public Member Functions

- **guideBoxDialog** (const string &pipeline, QWidget ∗parent=0)

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

- gui/include/cameraImageScene.h

9.356 Ui::guideBoxDialog Class Reference

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

- gui/build/ui_guideBoxDlg.h

9.357 GuideLoop Class Reference

Public Member Functions

- **GuideLoop** (Instrument &instrument, const TCSServerConfig &config, TCSDriver *driver, Metrology *master=NULL)
- void **start** (Message &m)
- void **start** (void)
- void **activate** (void)
- virtual void **deactivate** (void)
- void **set_fiducial** (double xpixel, double ypixel)
- void **clear_fiducial** (void)
- void **set_analysis_region** (int xmin, int xmax, int ymin, int ymax)
- vec2_t **get_fiducial** (void)
- AnalysisRegion **get_analysis_region** (bool do_exception=0)
- virtual void **apply_metrology** (Payload &p, const fptime &data_time, bool do_throw=0)
- virtual void **clear** (bool have_lock=0)
- double **get_orientation_radians** (void)
- const vec2_t & **get_plate_scale** (void)
- void **offset_fiducial** (vec2_t offset_asec, bool compensate=0, bool update_last_offset_time=1)
- void **offset_fiducial_equatorial** (vec2_t offset_asec, bool compensate=0, bool update_last_offset_time=1)

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

- tcs/metrology.h
- tcs/metrology.cpp

9.358 tcs_photometry::GuideRect Struct Reference

9.358.1 Detailed Description

Rectangular image region used for confining the location of the guide star

`#include <photometry.h>`

Public Member Functions

- **GuideRect** (const GuideRect &a)
- **GuideRect** (int xmin, int xmax, int ymin, int ymax)

Public Attributes

- int **m_xmin**
- int **m_xmax**
- int **m_ymin**
- int **m_ymax**

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

- het_lib/photometry/include/photometry.h

9.359 GuiderHandler Class Reference

Public Member Functions

- **GuiderHandler** (GuideLoop *l, const MessageHandlerDescriptor &m)

Protected Attributes

- GuideLoop * **guider_**

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

- tcs/metrology.h

9.360 tcs_photometry::GuiderMetrologyData Struct Reference

Public Attributes

- Object **obj_**
- PhotData **pd_**
- FitPars **fp_**

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

- het_lib/photometry/include/object.h

9.361 hetdex-dither::GuiderRelay Class Reference

9.361.1 Detailed Description

```
Class that relays calls to ``<guider>_*`` methods.

Example
-------

>>> r = GuiderRelay(pfip, 'Guider1')
>>> r.offset_fiducial(dx_ang=<dx>, dy_ang=<dy>)

is equivalent to

>>> pfip.Guide1_offset_fiducial(dx_ang=<dx>, dy_ang=<dy>)

Parameters
----------
tcs_subsystem : :class:`TCSSubSystem` instance
    tcs subsystem to which the guider belongs to
gname : string
    guider name. Can be either of ```'Guider1'`` or ```'Guider2'``

Attributes
----------
gname : string
    name of the guider
```

Public Member Functions

- def **__init__**
- def __getattr__

Public Attributes

- **gname**

9.361.2 Member Function Documentation

9.361.2.1 def hetdex-dither::GuiderRelay::__getattr__ (*self*, *name*)

```
Return the attribute matching ```'gname_name'``

Parameters
----------
name : string
    name of the method/attribute to get

Returns
-------
attribute/method
```

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

- scripting/scripts/hetdex-dither.py

9.362 GUIException Class Reference

Public Member Functions

- **GUIException** (tcsexcept::tcs_exception_t __gui, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.363 guiPreferences Class Reference

Public Attributes

- bool **m_bTempUnitsF**
- QColor **m_guiderCentroidFiducialColor**
- QColor **m_manualCentroidFiducialColor**
- QColor **m_guideBoxColor**
- QColor **m_manualCentroidBoxColor**
- QColor **m_instrumentFiducialColor**
- QColor **m_manualFiducialColor**
- QColor **m_measuringTriangleColor**
- QColor **m_measuringTextColor**
- QColor **m_markerTextColor**
- QColor **m_imageCursorColor**
- QColor **m_directionColor**
- QColor **m_WFSSubApertureOverlayColor**
- QColor **m_WFSM1OverlayColor**
- QColor **m_WFSPupilOverlayColor**
- QColor **m_WFSObscurationOverlayColor**
- QColor **m_WFSVectorsOverlayColor**
- QColor **m_guider1RangeColor**
- QColor **m_guider2RangeColor**
- QColor **m_wfs1RangeColor**
- QColor **m_wfs2RangeColor**

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

- gui/include/guiPreferences.h
- utility_gui/include/guiPreferences.h

9.364 guiSound Class Reference

Public Member Functions

- void **play** (string resourceName)

Protected Member Functions

- bool **openDevice** (unsigned int channels, unsigned int sampleRate, unsigned int bytesPerSample, _snd_pcm_format format)
- void **closeDevice** (bool bDrain=true)
- void **error** (const char ∗reason, const char ∗sndError)

Protected Attributes

- snd_pcm_t ∗ **m_deviceHandle**
- unsigned int **m_bufferSize**
- unsigned char ∗ **m_pBuffer**

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

- gui/include/guiSound.h
- gui/src/guiSound.cpp

9.365 liborchestrate::GuiSubsystem Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.366 handler_t Struct Reference

Public Attributes

- string **objectName**
- string **handlerName**
- string **probeName**

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

- gui/src/pfipControlTabUi.cpp
- gui/src/productionTabUi.cpp

9.367 HardwareElement Class Reference

Public Member Functions

- **HardwareElement** (const string &name, int idx, int id, bool debug=false)
- int **id** ()
- int **idx** ()
- bool **valid** ()
- const string & **hrl** () const

Protected Member Functions

- string **make_hrl** (const HardwareElement &parent, const string &token, int id)
- string **make_hrl** (const HardwareElement &parent, const string &token, const std::string id)
- virtual void **write** (FitsOutWrapper &f)

Protected Attributes

- int **id_**
- int **idx_**
- string **hrl_**
- bool **valid_**
- bool **debug_**

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

- camra/camra_hardware.h

9.368 headerChunk Class Reference

Public Attributes

- chunkHeader **header**
- char **sRiffType** [4]

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

- gui/include/guiSound.h

9.369 HelloFunctor Struct Reference

Public Member Functions

- void **operator()** (int arg)
- void **Member** (void)
- void **MemberArg** (std::string f)

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

- het_lib/tthread/test/hello.cpp

9.370 HelpHandler Class Reference

9.370.1 Detailed Description

Common message handler, associated with each sub-system Receiver, that replies with a list containing each MessageHandler registered with the Receiver.

`#include <builtin_handlers.h>`

Public Member Functions

- virtual void process (Message *message)

9.370.2 Member Function Documentation

9.370.2.1 virtual void HelpHandler::process (Message * *message*) [inline, virtual]

Message processing method that searches the given message for a "handler" parameter and, if found, responds with the usage statement associated with the specified message handler. If the "handler" parameter is not found in the given message, then the HelpHandler responds with the list of all MessageHandler objects associated with the Receiver.

Parameters:

handler - Optional requested MessageHandler.

machine_readable - Optional flag indicating whether the response should come in a machine readable format. See the Scripting documentation for a use of the machine_readable flag. Defaults to false.

Implements MessageHandler.

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

- het_lib/messaging/include/builtin_handlers.h

9.371 hetdex-dither::HetdexDitherError Class Reference

9.371.1 Detailed Description

```
Error raised by the module
```

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

- scripting/scripts/hetdex-dither.py

9.372 HETEarthOrientation Class Reference

9.372.1 Detailed Description

Singleton class wraps the IERS Earth orientation prediction (EOP) bulletin providing the EOP information for a reference UTC given as fptime.

`#include <het_earth_orientation.h>`

Static Public Member Functions

- static void configure (const std::string &ser7_path)
- static double lst (const fptime &reference=fptime::now())
- static std::string lst_str (const fptime &reference=fptime::now(), bool iso8601=true)
- static double **dut** (const fptime &reference=fptime::now())

  - static double pmx (const fptime &reference=fptime::now())
  - static double **pmy** (const fptime &reference=fptime::now())

9.372.2 Member Function Documentation

9.372.2.1 static void HETEarthOrientation::configure (const std::string & *ser7_path*) [static]

Configure the object given a ser7 file. Once the path is set it cannot change. The file at the given path may change but will not be reparsed until the current Julian Day rolls over.

9.372.2.2 double HETEarthOrientation::lst (const fptime & *reference* = `fptime::now()`) [static]

Returns local sidereal time in radians, for HET at the given time, defaulting to now.

9.372.2.3 string HETEarthOrientation::lst_str (const fptime & *reference* = `fptime::now()`, bool *iso8601* = `true`) [static]

Returns local sidereal time in hh:mm:ss string format, for HET at the given time, defaulting to now.

9.372.2.4 double HETEarthOrientation::pmx (const fptime & *reference* = `fptime::now()`) [static]

Polar motion coordinates in arcseconds x and y at the given time, defaulting to now.

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

- het_lib/common/include/het_earth_orientation.h
- het_lib/common/src/het_earth_orientation.cpp

9.373 HETEarthOrientationException Class Reference

Public Member Functions

- **HETEarthOrientationException** (tcsexcept::tcs_exception_t __het_earth_orientation, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.374 HETillumination Class Reference

Classes

- struct M1motion
- struct M4motion
- struct Spidermotion
- struct Trackermotion

Public Member Functions

- void **SetHETpupil** ()
- void **SetHETm1** ()
- void **SetHETtracker** ()
- void **SetHETtracker_noPlatform** ()
- void **SetHETm4** ()
- void **SetHETspider** ()
- void **NormalizeGeometries** (float scale, float dx, float dy, bool init=false)
- void **AffinegGeom** (PupilData *obj, float dx, float dy, float scale)
- void **AffinegGeom** (ObscGeom *obj, float dx, float dy, float scale, float dclock)
- void **AffinegGeom** (SHGrid *obj, float dx, float dy, float scale, float dclock)
- void **MoveM1Geom** ()
- void **MoveM4Geom** ()
- void **MoveSPDRGeom** ()
- void **MoveTRKGeom** ()
- void **HETM1Mask** ()
- void initIllumination (const int nsample_=256, bool pflag=false)

  initialize class and calculate illumination at center sphere

- float getIllumination (float t, float p, float r, float x, float y)

  calculate illumination for tracker angles theta, phi, rho and field position x,y. Units are degrees.

- void ComputeIllumination ()

  calculate the illumination at the currently set values of the geometries

- void ResetIllumination (bool pflag=false)

  reset geometries

Public Attributes

- int **nsample**
- float **Ttrk**
- float **Ptrk**
- float **Rtrk**
- float **Xfld**
- float **Yfld**
- float **Rtrkoffset**
- float **illumination**

- float **m_ref_illumination**
- wfslib **HET**
- PupilData **HETpupil**
- ObscGeom **HETtracker**
- ObscGeom **HETspider**
- ObscGeom **HETm4**
- SHGrid **HETM1**
- struct HETillumination::M1motion **M1rate**
- struct HETillumination::Trackermotion **Trkrate**
- struct HETillumination::Spidermotion **Spdrate**
- struct HETillumination::M4motion **M4rate**

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

- het_lib/illum_lib/include/illumination.h
- het_lib/illum_lib/src/illumination.cpp

9.375 hex_fit Class Reference

Public Types

- typedef double **value_type**

Public Member Functions

- **hex_fit** (itf_tsf &CT)
- double **operator()** (const double xdumb, const FVector< double, 6 > &p, FVector< double, 6 > &jac) const

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

- het_lib/coord_trans/src/itf_tsf.cpp

9.376 Home Class Reference

Command the dome to find its encoder home postion.

9.376.1 Detailed Description

Parameters:

none

`#include <legacyDomeSubSystem.h>`

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

- legacy/include/legacyDomeSubSystem.h

9.377 HomeProbes Class Reference

9.377.1 Detailed Description

Set each probe to point to a zero-point on the x or y axis. Guider1 and Guider2 be set to 0 in x. WFS1 and WFS2 will be set to 0 in y. If one or more of the probes is diabled by configuration, that probe will not be affected. The PFIPInterrupt() handler can be used to cancel this operation.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **HomeProbes** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.378 HomeShutter Class Reference

9.378.1 Detailed Description

Home the focal plane shutter.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **HomeShutter** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ACQShutterHandlers.cpp

9.379 HotPixel_t Class Reference

Public Member Functions

- **HotPixel_t** (int xValue, int yValue, int adjustValue)

Public Attributes

- int **x**
- int **y**
- int **adjust**

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

- pas/include/pasImageCleanerStandard.h

9.380 ICSConfig Class Reference

Static Public Member Functions

- static void **init** (int args, char **argv)
- static ICSQuery **query** (const std::string &q)
- static std::string & name (void)
- static std::string & **node** (void)
- static void **print_usage** (int argc, char **argv)
- static void **start_event_handling** (void)
- static bool **debug** (void)

9.380.1 Member Function Documentation

9.380.1.1 std::string & ICSConfig::name (void) [static]

Returns the name of this EventSource given at instantiation.

Reimplemented from EventSource.

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

- ics/ics.h
- ics/ics.cpp

9.381 ICSDaemon< S > Class Template Reference

template<typename S> class ICSDaemon< S >

Public Member Functions

- int **run** (const bool &help)
- void **register_handler** (MessageHandler *h)
- S & **subsystem** (void)

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

- ics/ics.h

9.382 ICSQuery Class Reference

Public Member Functions

- **ICSQuery** (const ICSQuery &q)
- **ICSQuery** (const std::string &q)
- bool **as_bool** (void)
- int **as_int** (void)
- unsigned **as_uint** (void)
- double **as_float** (void)

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

- ics/ics.h
- ics/ics.cpp

9.383 tthread::thread::id Class Reference

9.383.1 Detailed Description

Thread ID. The thread ID is a unique identifier for each thread.

See also:

thread::get_id()

`#include <tinythread.h>`

Public Member Functions

- id ()
- **id** (unsigned long int aId)
- **id** (const id &aId)
- id & **operator=** (const id &aId)

Friends

- bool **operator==** (const id &aId1, const id &aId2)
- bool **operator!=** (const id &aId1, const id &aId2)
- bool **operator**<**=** (const id &aId1, const id &aId2)
- bool **operator**< (const id &aId1, const id &aId2)
- bool **operator**>**=** (const id &aId1, const id &aId2)
- bool **operator**> (const id &aId1, const id &aId2)
- std::ostream & **operator**<< (std::ostream &os, const id &obj)

9.383.2 Constructor & Destructor Documentation

9.383.2.1 tthread::thread::id::id (void) `[inline]`

Default constructor. The default constructed ID is that of thread without a thread of execution.

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

- het_lib/tthread/include/tinythread.h

9.384 IglooData Class Reference

A class representing the main control values manipulated by the user on the Control tab. Also contains the alert list, for updating the dashboard alert system.

`#include <iglooStatusData.h>`

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)
- int **get_alert_messages** (alert_level_t level, QString &messages)

Public Attributes

- struct IglooData::status **m_status**
- IglooStatusFieldList * **p_fieldList**

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

- gui/include/iglooStatusData.h
- gui/src/iglooStatusData.cpp

9.385 IglooRegAlertList Class Reference

A collection of alert messages for the TCS GUI dashboard alert system.

9.385.1 Detailed Description

The alert messages are organized into lists by alert level, and all lists are populated by a call to initialize(), providing an event payload. For each parameter named "x.alert_level" ("x" being any string) found in that payload, an alert is created according to that parameter's value ("warning", "error" etc), and information about the alert is assembled from other parameters including x.desc, x.value, x.normal, x.limit_lower, and x.limit_upper.

`#include <iglooStatusData.h>`

Classes

- struct **alert**

Public Member Functions

- int initialize (Payload &payload)
- int get_messages (alert_level_t alert_level, QString &message)

9.385.2 Member Function Documentation

9.385.2.1 int IglooRegAlertList::get_messages (alert_level_t *alert_level*, QString & *message*)

Get the messages to display in the TCS GUI alerts window.

Parameters:

alert_level The category of alert, eg. warnings, errors, etc.

message Reference to receive the message list as a concatenated multiline string.

Returns:

The number of alert messages written to message.

9.385.2.2 int IglooRegAlertList::initialize (Payload & *payload*)

Clear and initialize the list of alerts from a given event payload.

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

- gui/include/iglooStatusData.h
- gui/src/iglooStatusData.cpp

9.386 IglooSensors Class Reference

9.386.1 Detailed Description

Igloo Sensors subsystem implementation class

`#include <igloo_sensors.h>`

Public Member Functions

- **IglooSensors** (const string &appName, IglooSensorsConfig *_p_config)
- int **start** ()
- void **stop** ()

Public Attributes

- IglooSensorsConnection **conn**

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

- igloo_sensors/igloo_sensors.h
- igloo_sensors/igloo_sensors.cpp

9.387 IglooSensorsConfig Class Reference

Public Member Functions

- **IglooSensorsConfig** (bool isNamed=false)
- virtual void print_summary ()

  Called prior to displaying options.

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- std::string **plc_hostname**
- int **plc_port**
- int **ui_poll_ms**
- int **arcv_poll_ms**

9.387.1 Member Function Documentation

9.387.1.1 virtual void IglooSensorsConfig::print_summary () [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.387.1.2 virtual void IglooSensorsConfig::set_options (OptionParser & *optionParser*) [inline, virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- igloo_sensors/igloo_sensors_config.h

9.388 IglooSensorsConnection Class Reference

Public Member Functions

- **IglooSensorsConnection** (IglooSensorsConfig *p_config)
- void **Get** (Message &reply, std::string name)
- void **Set** (Message &reply, std::string name, bool value)
- void **Toggle** (Message &reply, std::string name, bool value)

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

- igloo_sensors/hwcomm.h
- igloo_sensors/hwcomm.cpp

9.389 IglooSensorsException Class Reference

Public Member Functions

- **IglooSensorsException** (tcsexcept::tcs_exception_t __igloo_sensors, const std::string &w, const char ∗file, int line, const char ∗func)

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

- het_lib/common/include/tcs_exception.h

9.390 IglooSensorsHandler Class Reference

Public Member Functions

- **IglooSensorsHandler** (IglooSensors &igsen, const MessageHandlerDescriptor &m)

Protected Attributes

- IglooSensors & **igsen**

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

- igloo_sensors/igloo_sensors.h
- igloo_sensors/igloo_sensors.cpp

9.391 IglooStatusField Struct Reference

A structure containing the data for a "field" represented by a row of widgets in the Status tab.

9.391.1 Detailed Description

A field is populated from multiple values in the igloo_sensors UI event, with names in a dotted format to represent a hierarchical organization in a transport mechanism that doesn't support it directly. That organization is regenerated by creating instances of this structure, managed by IglooStatusFieldList.

`#include <iglooStatusData.h>`

Public Attributes

- std::string **label**
- std::string **desc**
- std::string **value**
- std::string **val_type**
- std::string **normal**
- std::string **limit_upper**
- std::string **limit_lower**
- bool **alarm_current**
- bool **alarm_latched**
- bool **alarm_enabled**

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

- gui/include/iglooStatusData.h

9.392 IglooStatusFieldList Class Reference

A class wrapping a map of IglooStatusField instances, storing data for a dynamic list of widgets.

9.392.1 Detailed Description

This class exists primarily for its constructor, which accepts a Payload (a UI event from the igloo_sensors subsystem) and populates the data structures from the parameters in that Payload.

`#include <iglooStatusData.h>`

Public Member Functions

- **IglooStatusFieldList** (Payload &payload)
- IglooStatusFieldListIterator **begin** (void)
- IglooStatusFieldListIterator **end** (void)
- IglooStatusFieldListIterator **find** (const std::string &label)

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

- gui/include/iglooStatusData.h
- gui/src/iglooStatusData.cpp

9.393 iglooStatusFieldWidgetSet Class Reference

A class representing a row of dynamic widgets pertaining to a particular "field" in the igloo_sensors subsystem interface. Each instance provides the GUI representation of an instance of IglooStatusField.

`#include <iglooTabUi.h>`

Signals

- void **SetRegister** (std::string, std::string, bool)

Public Member Functions

- iglooStatusFieldWidgetSet (iglooTabUi &parent, QBoxLayout *parentLayout, const std::string &reglabel)

  Create a row of widgets to insert into the dynamic list Creates an hbox and creates widgets inside it. Inserts the hbox into parentLayout.

- void update (IglooStatusField &field)

  Update the widgets with the contents of the given IglooStatusField.

- void **set_disabled** (void)
- bool **operator==** (const iglooStatusFieldWidgetSet &rhs)
- bool **operator==** (const std::string rhs)
- const std::string & **label** (void)

9.393.1 Constructor & Destructor Documentation

9.393.1.1 iglooStatusFieldWidgetSet::iglooStatusFieldWidgetSet (iglooTabUi & *parent*, QBoxLayout * *parentLayout*, const std::string & *reglabel*)

Parameters:

parent The iglooTabUi instance in use

parentLayout The layout (presumably a QVBoxLayout) which will receive the hbox and row of widgets

reglabel The text string describing this field, derived from the parameters in the UI event payload

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

- gui/include/iglooTabUi.h
- gui/build/moc_iglooTabUi.cpp
- gui/src/iglooTabUi.cpp

9.394 Ui::iglooTab Class Reference

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

- gui/build/ui_iglooTab.h

9.395 iglooTabUi Class Reference

Class representing the Igloo tab for TCS GUI.

`#include <iglooTabUi.h>`

Signals

- void **SetRegister** (std::string, std::string, bool)
- void **ForceHeartbeat** (std::string)

Public Member Functions

- **iglooTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

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

- gui/include/iglooTabUi.h
- gui/build/moc_iglooTabUi.cpp
- gui/src/iglooTabUi.cpp

9.396 ExposureInfo::Image Struct Reference

Public Attributes

- unsigned int **widthInPixels**
- unsigned int **heightInPixels**
- unsigned int **sizeInPixels**
- bool **bCleanedImageValid**
- MArray< float, 3 > * **pRawImage**
- MArray< float, 2 > * **pCleanedImage**
- MArray< float, 2 > * **pRedImage**
- MArray< float, 2 > * **pGreenImage**
- MArray< float, 2 > * **pBlueImage**
- MArray< float, 3 > * **pCoAddedImage**

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

- pas/include/Camera.h

9.397 Image Class Reference

Simple 2D BGR image class.

Public Member Functions

- Image (int aWidth, int aHeight)

  Create a new image with the dimensions aWidth x aHeight.

- void WriteToTGAFile (const char *aFileName)

  Write the image to a TGA file.

- Pixel & **operator[ ]** (const int idx)
- int **Width** () const
- int **Height** () const

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

- het_lib/tthread/test/fractal.cpp

9.398 ImageCleaner Class Reference

Public Member Functions

- **ImageCleaner** (PipelineConfig *pConfig)
- virtual bool **clean** (const int filterIndex, const CleaningBiasMode_t biasMode, ExposureInfo *pExposureInfo)=0
- virtual void **initialize** (void)

Protected Attributes

- PipelineConfig * **m_pConfig**

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

- pas/include/pasImageCleaner.h

9.399 ImageData Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar ∗memoryBlock=NULL, wfsData ∗additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar ∗∗memoryBlock=NULL, wfsData ∗additionalData=NULL)
- virtual void **newData** (int size)
- void **newFrame** (int size)
- virtual void **deleteData** (void)
- virtual void **deleteFrame** (void)
- void **CopyImage** (uint width, uint height, float wfs_gain, const float ∗img, const ImageMask &imgmask)
- void **CopyImage** (uint width, uint height, float wfs_gain, const double ∗img, const ImageMask &imgmask)

Public Attributes

- uint **Width**
- uint **Height**
- char **timestamp** [128]
- int **framenum**
- int **bitpix**
- uint **Npix**
- double ∗ **Data**
- uchar ∗ **Frame**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.400 pasImagePipelineSubSystem::ImageHandlerData Struct Reference

Public Attributes

- tthread::mutex **Mutex**
- tthread::condition_variable **DataReady**
- int **HandlerToken**
- std::map< int, bool > **SendImagesRunning**
- int **Count**
- int **LastCount**
- Camera_t **CameraType**
- std::string **CameraTypeName**
- FitsOutWrapper * **pFITSFile**
- uint8_t * **pFITSFileBuffer**
- size_t **FITSFileBufferSize**
- size_t **wfs_shgridSize**
- unsigned char * **wfs_shgrid**
- size_t **wfs_m1arraySize**
- unsigned char * **wfs_m1array**
- size_t **wfs_pupilSize**
- unsigned char * **wfs_pupil**
- size_t **wfs_hetobscSize**
- unsigned char * **wfs_hetobsc**
- size_t **wfs_shtrkdatSize**
- unsigned char * **wfs_shtrkdat**

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

- pas/include/pasImagePipelineSubSystem.h

9.401 ImageMask Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar ∗memoryBlock=NULL, wfsData ∗additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar ∗∗memoryBlock=NULL, wfsData ∗additionalData=NULL)
- virtual void **newData** (int size)
- virtual void **deleteData** (void)

Public Attributes

- uint **Npix**
- ushort ∗ **index**
- bool ∗ **mask**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.402 ImagePipelineGetExposureTimeHandlerClass Class Reference

Public Member Functions

- **ImagePipelineGetExposureTimeHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.402.1 Member Function Documentation

9.402.1.1 virtual void ImagePipelineGetExposureTimeHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.403 ImagePipelineGetGainHandlerClass Class Reference

Public Member Functions

- **ImagePipelineGetGainHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.403.1 Member Function Documentation

9.403.1.1 virtual void ImagePipelineGetGainHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.404 ImagePipelineGetOverScanHandlerClass Class Reference

Public Member Functions

- **ImagePipelineGetOverScanHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.404.1 Member Function Documentation

9.404.1.1 virtual void ImagePipelineGetOverScanHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.405 ImagePipelineGetPhotPhotometryOptionsHandlerClass Class Reference

Public Member Functions

- **ImagePipelineGetPhotPhotometryOptionsHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.405.1 Member Function Documentation

9.405.1.1 virtual void ImagePipelineGetPhotPhotometryOptionsHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.406 ImagePipelineGetRegionOfInterestHandlerClass Class Reference

Public Member Functions

- **ImagePipelineGetRegionOfInterestHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.406.1 Member Function Documentation

9.406.1.1 virtual void ImagePipelineGetRegionOfInterestHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.407 ImagePipelineGetWFSPhotometryOptionsHandlerClass Class Reference

Public Member Functions

- **ImagePipelineGetWFSPhotometryOptionsHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.407.1 Member Function Documentation

9.407.1.1 virtual void ImagePipelineGetWFSPhotometryOptionsHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.408 ImagePipelineSetCleanImagesHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetCleanImagesHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.408.1 Member Function Documentation

9.408.1.1 virtual void ImagePipelineSetCleanImagesHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.409 ImagePipelineSetCleaningBiasTypeHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetCleaningBiasTypeHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.409.1 Member Function Documentation

9.409.1.1 virtual void ImagePipelineSetCleaningBiasTypeHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.410 ImagePipelineSetFilterExposureTimeFactorHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetFilterExposureTimeFactorHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.410.1 Member Function Documentation

9.410.1.1 virtual void ImagePipelineSetFilterExposureTimeFactorHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.411 ImagePipelineSetFilterSequenceHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetFilterSequenceHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.411.1 Member Function Documentation

9.411.1.1 virtual void ImagePipelineSetFilterSequenceHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.412 ImagePipelineSetFilterZeroPointHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetFilterZeroPointHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.412.1 Member Function Documentation

9.412.1.1 virtual void ImagePipelineSetFilterZeroPointHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.413 ImagePipelineSetObjectAndMagnitudesHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetObjectAndMagnitudesHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.413.1 Member Function Documentation

9.413.1.1 virtual void ImagePipelineSetObjectAndMagnitudesHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.414 ImagePipelineSetOverScanHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetOverScanHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.414.1 Member Function Documentation

9.414.1.1 virtual void ImagePipelineSetOverScanHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.415 ImagePipelineSetPhotometryOptionsHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetPhotometryOptionsHandlerClass** (SubSystem *subSystem)
- virtual void process (Message *pMessage)

9.415.1 Member Function Documentation

9.415.1.1 void ImagePipelineSetPhotometryOptionsHandlerClass::process (Message * *message*) [virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h
- pas/src/pasImagePipelineSubSystem.cpp

9.416 ImagePipelineSetPhotPhotometryOptionsHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetPhotPhotometryOptionsHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.416.1 Member Function Documentation

9.416.1.1 virtual void ImagePipelineSetPhotPhotometryOptionsHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.417 ImagePipelineSetProcessingTypeHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetProcessingTypeHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.417.1 Member Function Documentation

9.417.1.1 virtual void ImagePipelineSetProcessingTypeHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.418 ImagePipelineSetWFSPhotometryOptionsHandlerClass Class Reference

Public Member Functions

- **ImagePipelineSetWFSPhotometryOptionsHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.418.1 Member Function Documentation

9.418.1.1 virtual void ImagePipelineSetWFSPhotometryOptionsHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.419 ImagePipelineStartSendingImagesHandlerClass Class Reference

Public Member Functions

- **ImagePipelineStartSendingImagesHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.419.1 Member Function Documentation

9.419.1.1 virtual void ImagePipelineStartSendingImagesHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasImagePipelineSubSystem.h

9.420 ImagePipelineStatus Class Reference

Public Attributes

- string **name**
- double **cycleTime**
- int **coAddedImages**
- bool **pipelineRunning**
- bool **cleaningConfigured**
- bool **cleaningImages**
- bool **processingConfigured**
- bool **processingImages**
- int **processingMode**
- bool **storingConfigured**
- bool **storingImages**
- int **type**
- string **typeName**
- int **pixelFormat**
- int **imageType**
- int **binningWidth**
- int **binningHeight**
- int **roiX**
- int **roiY**
- int **roiWidth**
- int **roiHeight**
- double **exposureTime**
- int **flushes**
- int **imageWidthInPixels**
- int **imageHeightInPixels**
- int **filterCount**
- string **filterNames**
- string **filterSequence**
- int **filterSequenceLength**
- string **filterExposureFactors**
- int **filterZeroPoints**
- int **filterCurrentFilter**
- int **sensorWidth**
- int **sensorHeight**
- int **sensorBitsPerPixel**
- double **sensorGain**
- bool **sensorTemperatureSupported**
- double **sensorTemperature**
- bool **sensorTargetTemperatureSupported**
- double **sensorTargetTemperature**
- double **plateScaleX**
- double **plateScaleY**

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

- gui/include/tcsGui.h

9.421 ImageProcessor Class Reference

Base Image Processor class.

`#include <ImageProcessor.h>`

Public Member Functions

- **ImageProcessor** (TCSConfig *pConfig)
- virtual bool **process** (ProcessingMode_t processingMode, ExposureInfo *pExposureInfo, pasImagePipelineSubSystem *pSubSystem, Payload &event)=0
- virtual void **getPhotOptions** (tcs_photometry::ConfigureParams &configParams)
- virtual void **getPhotOptionsToPayload** (Payload &payload)
- virtual void **setPhotOptions** (const tcs_photometry::ConfigureParams &configParams)
- virtual void **getGraphicOverlayData** (unsigned char **wfs_shgrid, size_t &wfs_shgridSize, unsigned char **wfs_m1array, size_t &wfs_m1arraySize, unsigned char **wfs_pupil, size_t &wfs_pupilSize, unsigned char **wfs_hetobsc, size_t &wfs_hetobscSize, unsigned char **wfs_shtrkdat, size_t &wfs_shtrkdatSize)

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

- pas/include/ImageProcessor.h

9.422 imageRect< Type > Class Template Reference

template<typename Type> class imageRect< Type >

Public Member Functions

- **imageRect** (double xMin, double yMin, double xMax, double yMax)
- Type **xMin** ()
- Type **xMax** ()
- Type **yMin** ()
- Type **yMax** ()
- Type **width** ()
- Type **height** ()
- imageRect **displayToExternal** (Type imageHeight)
- imageRect **externalToDisplay** (Type imageHeight)

Protected Attributes

- Type **m_xMin**
- Type **m_yMin**
- Type **m_xMax**
- Type **m_yMax**

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

- gui/include/tcsGui.h

9.423 index_time Class Reference

Public Member Functions

- **index_time** (const double &other)

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

- het_lib/common/include/fptime.h

9.424 init_t Struct Reference

Public Attributes

- double **coef**
- double **coef1**
- double **tsi**
- double **s4**
- double **unused_a3ovk2**
- double **eta**

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

- tcs/traj_comp/NORADlib/norad_in.h

9.425 initTMCS Class Reference

9.425.1 Detailed Description

Initializes the TMCS hardware.

See also:

tracker_init

`#include <tracker_handler_doc.h>`

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

- tracker/tracker_handler_doc.h

9.426 tracker_handlers::initTMCS Class Reference

Public Member Functions

- **initTMCS** (TrackerDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.426.1 Member Function Documentation

9.426.1.1 void tracker_handlers::initTMCS::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tracker/tracker_handlers.h

9.427 inject_metrology Class Reference

9.427.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **inject_metrology** (MetrologyLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.427.2 Member Function Documentation

9.427.2.1 void inject_metrology::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.428 Instrument Class Reference

Public Member Functions

- **Instrument** (inst::id id)
- void **get_current_position** (instrument_position &sky, instrument_position &fpa, eq_coords &ceq)
- void **set_current_position** (const instrument_position &sky, const instrument_position &fpa, const double &orientation)
- double **get_current_orientation** (void)
- std::string name ()
- bool **is_movable** ()
- void **set_absolute_commanded_position** (const eq_coords &commanded)
- void **set_relative_target_position** (const instrument_position &relative)
- bool **offset_relative_target_position** (const vec2_t &offset)
- bool **is_target_set** (void)
- instrument_position **get_target_position** (TCSDriver &driver_)
- void **reset** (void)
- void **set_filter** (PFIPClient &pfip, int filter)
- inst::id **id** ()
- bool **enabled** (void)
- bool **active** (void)
- void **activate** (void)
- void **enable** (void)
- void **deactivate** (void)
- void **disable** (void)
- void **emit_pointing_event** (TCSDriver &d)
- void **build_pointing_payload** (Payload &p, TCSDriver &d)

9.428.1 Member Function Documentation

9.428.1.1 std::string Instrument::name (void)

Returns the name of this EventSource given at instantiation.

Reimplemented from EventSource.

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

- tcs/instrument.h
- tcs/instrument.cpp

9.429 InterruptHandler Class Reference

9.429.1 Detailed Description

MessageHandler that is used to instantiate an interrupt handler for each Receiver instance. The InterruptHandler expects a singal "id" integer parameter that represents the message identifier associated with the MessageHandler instance that will be interrupted.

If the identifier given is not recognized by the Receiver, an error is returned to the remote caller. Otherwise, the Message::interupt() method of the effective Message is called.

`#include <builtin_handlers.h>`

Public Member Functions

- virtual void process (Message *message)

9.429.2 Member Function Documentation

9.429.2.1 virtual void InterruptHandler::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- het_lib/messaging/include/builtin_handlers.h

9.430 fixed_point::fixed_internal::ioflags Class Reference

9.430.1 Detailed Description

save flags of stream and restore upon destruction.

`#include <fixed.h>`

Public Member Functions

- **ioflags** (std::basic_ios< char > &stream)

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

- het_lib/common/include/fixed.h

9.431 IssuePMACCmd Class Reference

yyy

9.431.1 Detailed Description

Issue a command to the shutter PMAC card.

Parameters:

cmd <string> - PMAC command

cmd <string> - PMAC command

```
#include <legacyDomeSubSystem.h>
```

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

- legacy/include/legacyDomeSubSystem.h

9.432 itf_coordinate Struct Reference

9.432.1 Detailed Description

Class representing a point in the Ideal Tracker Frame.

`#include <tcs_coordinates.h>`

Public Member Functions

- **itf_coordinate** (double z)
- **itf_coordinate** (double i[6])
- **itf_coordinate** (const fptime &t, const double &x, const double &y, const double &z, const double &rho, const double &theta, const double &phi)
- **itf_coordinate** (const tracker_coordinate &t)
- virtual void **apply_parms** (Payload &p) const

Public Attributes

- double **x_**
- double **y_**
- double **z_**
- double **rho_**
- double **theta_**
- double **phi_**

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

- het_lib/common/include/tcs_coordinates.h
- het_lib/wfs_lib/include/reducewfs.h
- het_lib/common/src/tcs_coordinates.cpp

9.433 itf_tsf Class Reference

9.433.1 Detailed Description

Tracker frame For going between ITF and tracker sensor frame

`#include <coords.h>`

Public Member Functions

- **itf_tsf** (bool structureMM, bool tiptiltMM, bool trackingMM)
- ltl::FVector< double, 6 > **CalcHex** (const ltl::FVector< double, 6 > &ITFcoms, ltl::FVector< double, 6 > &StrutCorrection, ltl::FVector< double, 6 > &Fstruts, const bool inverseCalc) const
- ltl::FVector< double, 6 > **CalcITF** (const ltl::FVector< double, 10 > &SensorIn, const ltl::FVector< double, 6 > &ITFguess, ltl::FVector< double, 6 > &ITFerr)
- ltl::FVector< double, 6 > **LegLenComp** (const ltl::FVector< double, 6 > &ITF_Cmds, const bool inverseCalc) const

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

- het_lib/coord_trans/include/coords.h
- het_lib/coord_trans/src/itf_tsf.cpp

9.434 julian_day Class Reference

Public Member Functions

- **julian_day** (const double &other)

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

- het_lib/common/include/fptime.h

9.435 tcsdb::key_descriptor Class Reference

Public Member Functions

- def **__init__**
- def **__call__**
- def **add_attr**

Public Attributes

- **cursor_**
- **system_**
- **source_**
- **key_**

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

- scripting/pytcs/build/lib.linux-x86_64-2.7/tcsdb.py

9.436 pytcs::tcsdb::key_descriptor Class Reference

Public Member Functions

- def **__init__**
- def **__call__**
- def **add_attr**

Public Attributes

- **cursor_**
- **system_**
- **source_**
- **key_**

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

- scripting/pytcs/tcsdb.py

9.437 legacy_ForceStatusEvents Class Reference

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

- legacy/include/legacy_handler_doc.h

9.438 legacyConfig Class Reference

Public Member Functions

- **legacyConfig** (bool isNamed=false)
- void **parseSubSystemConfigs** (void)
- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

- virtual void print_summary (void)

  Called prior to displaying options.

Public Attributes

- DomeConfig * **m_pDomeConfig**
- ShutterConfig * **m_pShutterConfig**
- StructureConfig * **m_pStructureConfig**
- WeatherConfig * **m_pWeatherConfig**

9.438.1 Member Function Documentation

9.438.1.1 virtual void legacyConfig::print_summary (void) [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.438.1.2 void legacyConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- legacy/include/legacyConfig.h
- legacy/src/legacyConfig.cpp

9.439 liborchestrate::LegacyDomeSimulator Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.440 legacyDomeSubSystem Class Reference

Provides access to the legacy Dome server.

9.440.1 Detailed Description

The Dome Sub-System implements a "work" loop to retrieve status from the legacy SAMS server on a configurable cycle.

The status data retrieved is:

- Position
- Target
- Velocity
- Direction
- Moving
- PMACComms
- LocalRemote
- EMO

The Dome Sub-System responds to the following commands from senders:

- OnLine
-
- Abort
- GetPosition
- Home
- IssuePMACCmd
- MoveTo
- MoveToCCAS
- SetDirection
- SetPower
- SetVelocity
- Reset
- Status
- Stop
- Stow
- Test

```
#include <legacyDomeSubSystem.h>
```

Classes

- struct status

Public Types

- enum **Cmd_t** {
  ABORT, **DIRECTION**, **HOME**, **MOVE_TO**,
  MOVE_TO_CCAS, **PMAC_CMD**, **POSITION**, **POWER**,
  RESET, **STATUS**, **STATUS_VELOCITY**, **STOP**,
  STOW, **TEST**, **VELOCITY** }
- enum **Dir_t** { **CCW** = -1, **SHORTEST**, **CW** }

Public Member Functions

- **legacyDomeSubSystem** (DomeConfig *pConfig)
- virtual void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- virtual void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- **HANDLER_DECL** (Dome, OnLine)
- **HANDLER_DECL** (Dome, Abort)
- **HANDLER_DECL** (Dome, GetPosition)
- **HANDLER_DECL** (Dome, Home)
- virtual bool **IssuePMACCmdHandler** (Message *pMessage, Message **ppReply, string cmd)
- virtual bool **MoveToHandler** (Message *pMessage, Message **ppReply, double target)
- **HANDLER_DECL** (Dome, MoveToCCAS)
- **HANDLER_DECL** (Dome, Reset)
- virtual bool **SetDirectionHandler** (Message *pMessage, Message **ppReply, string direction)
- virtual bool **SetPowerHandler** (Message *pMessage, Message **ppReply, string state)
- virtual bool **SetVelocityHandler** (Message *pMessage, Message **ppReply, string velocity)
- virtual bool **StatusHandler** (Message *pMessage, Message **ppReply, bool update_state)
- **HANDLER_DECL** (Dome, Stop)
- **HANDLER_DECL** (Dome, Stow)
- virtual bool **TestHandler** (Message *pMessage, Message **ppReply, string state)
- virtual void **clearActive** (Message *pMessage)

Public Attributes

- class DomeIssuePMACCmdHandlerClass * **m_IssuePMACCmdHandler**
- class DomeMoveToHandlerClass * **m_MoveToHandler**
- class DomeSetDirectionHandlerClass * **m_SetDirectionHandler**
- class DomeSetPowerHandlerClass * **m_SetPowerHandler**
- class DomeSetVelocityHandlerClass * **m_SetVelocityHandler**
- class DomeStatusHandlerClass * **m_StatusHandler**
- class DomeTestHandlerClass * **m_TestHandler**

Protected Member Functions

- bool **moveTo** (Cmd_t cmd, double dTarget)
- double **computeMove** (double target, double position, Dir_t direction)
- void **waitPosition** (double target)
- void **populateStatusPayload** (Payload &payload)
- void **setupCmds** (void)
- virtual bool **initialize** (void)
- virtual double **statusWorker** (bool bForceEvent=false)
- virtual bool **checkReply** (int nCmd, unsigned int replyLength, unsigned char *replyString)
- virtual bool **getStatus** (bool retry=false, bool doException=false, bool extraBool=false)
- virtual void **checkAndSetActive** (Message *pMessage)

Protected Attributes

- DomeConfig * **m_pConfig**
- map< Cmd_t, string > **m_cmd**
- tthread::mutex **m_MutexMoveStopInfo**
- bool **m_bStopIssued**
- status **m_status**
- status **m_lastStatus**

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

- legacy/include/legacyDomeSubSystem.h
- legacy/src/legacyDomeSubSystem.cpp

9.441 LegacyException Class Reference

Public Member Functions

- **LegacyException** (tcsexcept::tcs_exception_t __legacy, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.442 legacyServer Class Reference

Public Member Functions

- **legacyServer** (const string &appName, legacyConfig *pConfig)
- virtual int **start** (void)
- virtual void **stop** (void)
- **SERVER_HANDLER_DECL** (ForceStatusEvents)

Protected Member Functions

- void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- void **SetName** (string name)
- const char * **GetName** (void)
- void **SetDebug** (bool on)

Protected Attributes

- string **m_Name**
- bool **m_Debug**
- legacyConfig * **m_Config**
- Receiver * **m_Receiver**
- legacyDomeSubSystem * **m_DomeSubSystem**
- legacyShutterSubSystem * **m_ShutterSubSystem**
- legacyStructureSubSystem * **m_StructureSubSystem**
- legacyWeatherSubSystem * **m_WeatherSubSystem**

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

- legacy/include/legacyServer.h
- legacy/src/legacyServer.cpp

9.443 liborchestrate::LegacyShutterSimulator Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.444 legacyShutterSubSystem Class Reference

Provides access to the legacy Shutter server.

9.444.1 Detailed Description

The Shutter Sub-System implements a "work" loop to retrieve status from the legacy SAMS server on a configurable cycle.

The status data retrieved is:

- Position
- Target
- Moving
- PMACComms
- ErrorNum
- LocalRemote

The Shutter Sub-System responds to the following commands from senders:

- OnLine
-
- Abort
- Close
- IssuePMACCmd
- Open
- Reset
- SetPower
- Status
- Stop

```
#include <legacyShutterSubSystem.h>
```

Classes

- struct status

Public Types

- enum **Cmd_t** {
  ABORT, **CMD**, **PMAC_CMD**, **POWER**,
  RESET, **STATUS**, **STOP** }

Public Member Functions

- **legacyShutterSubSystem** (ShutterConfig *pConfig)
- virtual void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- virtual void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- **HANDLER_DECL** (Shutter, OnLine)
- **HANDLER_DECL** (Shutter, Abort)
- **HANDLER_DECL** (Shutter, Close)
- virtual bool **IssuePMACCmdHandler** (Message *pMessage, Message **ppReply, string cmd)
- **HANDLER_DECL** (Shutter, Open)
- **HANDLER_DECL** (Shutter, Reset)
- virtual bool **SetPowerHandler** (Message *pMessage, Message **ppReply, string state)
- virtual bool **StatusHandler** (Message *pMessage, Message **ppReply, bool update_state)
- **HANDLER_DECL** (Shutter, Stop)
- virtual void **clearActive** (Message *pMessage)

Public Attributes

- class ShutterIssuePMACCmdHandlerClass * **m_IssuePMACCmdHandler**
- class ShutterSetPowerHandlerClass * **m_SetPowerHandler**
- class ShutterStatusHandlerClass * **m_StatusHandler**

Protected Member Functions

- bool **waitPosition** (double target)
- bool **waitState** (string target)
- void **populateStatusPayload** (Payload &payload)
- void **setupCmds** (void)
- virtual bool **initialize** (void)
- virtual double **statusWorker** (bool bForceEvent=false)
- virtual bool **checkReply** (int nCmd, unsigned int replyLength, unsigned char *replyString)
- virtual bool **getStatus** (bool retry=false, bool doException=false, bool extraBool=false)
- virtual void **checkAndSetActive** (Message *pMessage)

Protected Attributes

- ShutterConfig * **m_pConfig**
- map< Cmd_t, string > **m_cmd**
- tthread::mutex **m_MutexMoveStopInfo**
- bool **m_bStopIssued**
- status **m_status**
- status **m_lastStatus**

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

- legacy/include/legacyShutterSubSystem.h
- legacy/src/legacyShutterSubSystem.cpp

9.445 liborchestrate::LegacyStructureSimulator Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.446 legacyStructureSubSystem Class Reference

Provides access to the legacy Structure server.

9.446.1 Detailed Description

The Structure Sub-System implements a "work" loop to retrieve status from the legacy Structure server on a configurable cycle.

The status data retrieved and sent back for a Status command is:

- Azimuth_deg
- Elevation_deg
- InPosition
- HaveComms
- InControl
- Moving
- NewFault
- nFault
- NAKMsg_str

The Structure Sub-System responds to the following commands from senders:

- OnLine
- MoveTo
- MoveToStructAz
- MoveToCCAS
- Status
- Stop
- Stow

```
#include <legacyStructureSubSystem.h>
```

Classes

- struct status

Public Types

- enum **Cmd_t** {
  FOOT, **MAINTENANCE**, **MOVE_TO_TRUE_AZ**, **MOVE_TO_STRUCT_AZ**,
  MOVE_TO_CCAS, **LINK**, **RELEASE_CONTROL**, **STATUS**,
  STOP_MOVE, **STOW**, **TAKE_CONTROL** }

Public Member Functions

- **legacyStructureSubSystem** (StructureConfig *pConfig)
- virtual void **stop** (bool relinquishControl=false)
- virtual void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- virtual void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- **HANDLER_DECL** (Structure, OnLine)
- virtual bool **MoveToTrueAzHandler** (Message *pMessage, Message **ppReply, double target)
- virtual bool **MoveToStructAzHandler** (Message *pMessage, Message **ppReply, double target)
- **HANDLER_DECL** (Structure, MoveToCCAS)
- virtual bool **StatusHandler** (Message *pMessage, Message **ppReply, bool update_state)
- **HANDLER_DECL** (Structure, Stop)
- **HANDLER_DECL** (Structure, Stow)
- virtual void **clearActive** (Message *pMessage)

Public Attributes

- class StructureMoveToTrueAzHandlerClass * **m_MoveToTrueAzHandler**
- class StructureMoveToStructAzHandlerClass * **m_MoveToStructAzHandler**
- class StructureStatusHandlerClass * **m_StatusHandler**

Protected Member Functions

- double **computeMove** (double position1, double position2)
- void **moveTo** (Cmd_t cmd, double trueAzTarget, double structAzTarget)
- bool **waitPosition** (double trueAzTarget, double structAzTarget)
- void **populateStatusPayload** (Payload &pPayload)
- bool **releaseControl** (void)
- bool **setTimeout** (double timeout)
- bool **stopMove** (void)
- bool **takeControl** (void)
- void **setupCmds** (void)
- virtual bool **initialize** (void)
- virtual double **statusWorker** (bool bForceEvent=false)
- virtual bool **checkReply** (int nCmd, unsigned int replyLength, unsigned char *replyString)
- virtual bool **getStatus** (bool retry=false, bool doException=false, bool getFootHeights=false)
- virtual void **checkAndSetActive** (Message *pMessage)

Protected Attributes

- StructureConfig * **m_pConfig**
- std::map< Cmd_t, std::string > **m_cmd**
- std::string **m_footCmd**
- tthread::mutex **m_MutexMoveStopInfo**
- bool **m_bStopIssued**
- timespec **m_lastStatusUpdate**
- status **m_status**
- status **m_lastStatus**

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

- legacy/include/legacyStructureSubSystem.h
- legacy/src/legacyStructureSubSystem.cpp

9.447 liborchestrate::LegacySubsystem Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.448 Ui::legacyTab Class Reference

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

- gui/build/ui_legacyTab.h

9.449 legacyTabUi Class Reference

Signals

- void **ForceHeartbeat** (std::string)
- void **ForceLegacyStatusEvents** (void)
- void **ForceLoopStateEvent** (std::string)
- void **ForcePipelineStatusEvent** (std::string)
- void **StartPipeline** (std::string)
- void **StopPipeline** (std::string)
- void **StartMetrologyLoop** (std::string)
- void **StopMetrologyLoop** (std::string)
- void **ActivateMetrologyLoop** (std::string)
- void **DeactivateMetrologyLoop** (std::string)
- void **SetFocus** (double)
- void **MoveDomeTo** (double)
- void **SetDomeVelocityTo** (std::string)
- void **MoveStructureToTrueAz** (double)
- void **MoveStructureToStructAz** (double)
- void **SetDomeDirTo** (std::string)
- void **SetIntervalSpeed** (std::string, double)
- void **EnableSACDMIMeasurements** (std::string, bool)
- void **EnableCCASDMIMeasurements** (std::string, bool)

Public Member Functions

- **legacyTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Protected Member Functions

- void **LoadVelocityValues** (void)

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

- gui/include/legacyTabUi.h
- gui/build/moc_legacyTabUi.cpp
- gui/src/legacyTabUi.cpp

9.450 legacyWeatherSubSystem Class Reference

Provides access to the legacy HET Weather server.

9.450.1 Detailed Description

The Weather Sub-System implements a "work" loop which retrieves data from the legacy HET Weather server on a configurable cycle.

The data retrieved is:

- GMT
- CURRENT_WARNS
- MASTER_WARNS
- TEMPERATURE
- HUMIDITY
- INT_DEW_POINT
- EXT_DEW_POINT
- PRESSURE
- WIND_DIR
- WIND_SPEED
- EXT_DUST_1
- EXT_DUST_2
- INT_DUST_1
- INT_DUST_2
- RAIN

The Weather Sub-System responds to the following commands from senders:

- OnLine
- GetData

```
#include <legacyWeatherSubSystem.h>
```

Classes

- struct statistic
- struct status

Public Types

- enum **Cmd_t** { **GET_DATA** }
- enum **Data_t** {
  GMT, **CURR_WARN**, **MASTER_WARN**, **TEMP**,
  HUMIDITY, **INT_DEW_POINT**, **EXT_DEW_POINT**, **PRESSURE**,
  WIND_DIR, **WIND_SPEED**, **EXT_DUST_1**, **EXT_DUST_2**,
  INT_DUST_1, **INT_DUST_2**, **RAIN**, **TRUSS**,
  TRUSS_AIR, **TRACKER**, **MIRROR** }

Public Member Functions

- **legacyWeatherSubSystem** (WeatherConfig *pConfig)
- virtual void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- virtual void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- **HANDLER_DECL** (Weather, OnLine)
- virtual bool **GetDataHandler** (Message *pMessage, Message **ppReply, bool update_state)

Public Attributes

- class WeatherGetDataHandlerClass * **m_GetDataHandler**

Protected Types

- typedef std::pair< Data_t, std::string > **DataItemLabel_t**

Protected Member Functions

- int **processGetDataReplyItem** (const Data_t replyItem, const std::string replyItemLabel, std::vector< std::string > &getDataTokens)
- void **setupStatusTokens** (void)
- bool **waitPosition** (double target)
- void **populateStatusPayload** (Payload &pPayload)
- void **populateStatusPayloadStats** (Payload &payload, const statistic *pStat)
- virtual bool **initialize** (void)
- virtual double **statusWorker** (bool bForceEvent=false)
- virtual bool **checkReply** (int nCmd, unsigned int replyLength, unsigned char *replyString)
- virtual bool **getStatus** (bool retry=false, bool doException=false, bool extraBool=false)

Protected Attributes

- WeatherConfig * **m_pConfig**
- std::map< std::string, DataItemLabel_t > **m_StatusToken**
- status **m_status**
- status **m_lastStatus**

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

- legacy/include/legacyWeatherSubSystem.h
- legacy/src/legacyWeatherSubSystem.cpp

9.451 LightSource Class Reference

Base Light Source class.

`#include <LightSource.h>`

Public Member Functions

- **LightSource** (TCSConfig *pConfig)
- virtual bool **powerOn** (bool bChangePowerLevel=0)=0
- virtual bool **powerOff** (void)=0
- virtual bool **isPowerOn** (void)=0
- virtual bool **setPowerHoldTime** (int holdTime)=0
- virtual bool **setPowerLevel** (double powerLevel)=0
- virtual bool **getPowerLevel** (double *pPowerLevel)=0
- virtual bool **atFullPower** (void)=0

Protected Attributes

- TCSSocket * **m_pSocket**

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

- pas/include/LightSource.h

9.452 LightSourceConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- string **TTLaser_address**
- double m_powerLevel

  Power level.

- double m_powerHoldTime_secs

  Power hold time.

- double m_powerMonitorCycleTime_secs

  Power Monitor cycle time.

- bool m_socketDebug

  Enable socket debugging.

9.452.1 Member Function Documentation

9.452.1.1 void LightSourceConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

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

- pas/include/pasConfig.h
- pas/src/pasConfig.cpp

9.453 linFit Class Reference

9.453.1 Detailed Description

This is a series of definitions to fit a data set to various polynomials. It uses the LTL fitting routine in lmfit.h

`#include <curveFits.h>`

Public Types

- typedef double **value_type**

Public Member Functions

- double **operator()** (const double &x, const ltl::FVector< double, 2 > &p, ltl::FVector< double, 2 > &jac) const

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

- het_lib/coord_trans/mount_model/curveFits.h

9.454 load_trajectory Class Reference

9.454.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **load_trajectory** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.454.2 Member Function Documentation

9.454.2.1 void load_trajectory::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.455 loadedUi Class Reference

Public Member Functions

- **loadedUi** (const char *eventSourceName, QWidget *pLoadedWidget, guiConfig *pConfig)
- QWidget * **getLoadedWidget** (void)
- guiConfig * **getConfig** (void)
- **loadedUi** (const char *eventSourceName, QWidget *pLoadedWidget, guiConfig *pConfig)
- QWidget * **getLoadedWidget** (void)
- guiConfig * **getConfig** (void)

Protected Slots

- void **parentTabChanged** (int index)
- void **parentTabChanged** (int index)

Protected Member Functions

- virtual void **updateUi** (void)
- virtual void **showDialogs** (bool=true)
- virtual void **updateUi** (void)
- virtual void **showDialogs** (bool=1)

Protected Attributes

- QWidget * **m_pLoadedWidget**
- guiConfig * **m_pConfig**

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

- gui/include/loadedUi.h
- utility_gui/include/loadedUi.h

9.456 loadMarkersDialog Class Reference

Public Member Functions

- void **loadData** (cameraImageTabUi::markerMap_t *pMarkerMap)

Protected Member Functions

- void **loadSettings** (void)
- void **saveSettings** (void)

Protected Attributes

- cameraImageTabUi::markerMap_t * **m_pMarkerMap**

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

- gui/include/loadMarkersDlg.h
- gui/src/loadMarkersDlg.cpp

9.457 Ui::loadMarkersDialog Class Reference

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

- gui/build/ui_loadMarkersDlg.h

9.458 tthread::lock_guard< T > Class Template Reference

9.458.1 Detailed Description

template<class T> class tthread::lock_guard< T >

Lock guard class. The constructor locks the mutex, and the destructor unlocks the mutex, so the mutex will automatically be unlocked when the lock guard goes out of scope. Example usage:

```
mutex m;
int counter;

void increment()
{
  lock_guard<mutex> guard(m);
  ++ counter;
}
```

`#include <tinythread.h>`

Public Types

- typedef T **mutex_type**

Public Member Functions

- lock_guard (mutex_type &aMutex)

  The constructor locks the mutex.

- ~lock_guard ()

  The destructor unlocks the mutex.

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

- het_lib/tthread/include/tinythread.h

9.459 LogClientException Class Reference

Public Member Functions

- **LogClientException** (tcsexcept::tcs_exception_t __logclient, const std::string &w, const char ∗file, int line, const char ∗func)

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

- het_lib/common/include/tcs_exception.h

9.460 Logger Class Reference

9.460.1 Detailed Description

The client-side logger class provides a common interface for logging and error reporting for HET. The client logger provides convenience macros for entering log messages. The macros (and the SendLogMessage method) send the log messages as an event and produce a formatted console message on stderr.

`#include <Logger.h>`

Public Types

- enum LogLevel {

  NONE = 0, **DEBUG** = 1, **INFO** = 2, **WARN** = 4,

  ERROR = 8, **FATAL** = 16, **ALARM** = 32 }

  Enumeration type for log severity type. Used by the logging macros.

Static Public Member Functions

- static void SendLogMessage (EventSource *e, LogLevel ll, const char *msg, const char *fileN, int lineN, const char *func)

  Submit a log message, for use by the logging macro.

- static int SendExternalLogMessage (EventSource *ev, Sender *s, LogLevel ll, const char *msg, const char *fileN, int lineN, const char *func)

  Submit a log message, for use by any external system wishing to log such as python scripts or 3rd party tools. This works by forwarding the log message to the tcs_log_relay subsystem.

- static std::string ll_to_string (LogLevel ll, bool for_event=false)

  A convenience routine that converts the log level enum value to a string. Since there are different log level string values for events and console log, use the bool for_event parameter set to true, if the logging level is generated for an event.

- static void fmtConsole (const char *ll, const char *msgTime, const char *msg, const char *fileName, std::string &outStr)

  Format a console-style message.

- static void makeFileLoc (const char *fileN, const char *funcN, int lineN, std::string &fileStr, std::string &fileLine)

  Create a string with line number and file name for log message display.

- static std::string **output_level** (const std::string &lev)
- static void **TurnOffLogToConsole** (void)
- static void **RetrySendingToRelay** (void)

9.460.2 Member Function Documentation

9.460.2.1 void Logger::fmtConsole (const char * *ll*, const char * *msgTime*, const char * *msg*, const char * *fileName*, std::string & *outStr*) [static]

Parameters:

in[ll: logging severity level

← ***msgTime,:*** timestamp of the message origination

← ***msg,:*** the formatted message string

← ***fileName,:*** a string containing the source file location of message

→ ***outStr,:*** the formatted output string

9.460.2.2 void Logger::makeFileLoc (const char * *fileN*, const char * *funcN*, int *lineN*, std::string & *fileStr*, std::string & *fileLine*) [static]

Parameters:

← file name

← line number

→ location formatted for event

→ location formatted for console

9.460.2.3 int Logger::SendExternalLogMessage (EventSource * *ev*, Sender * *s*, LogLevel *ll*, const char * *msg*, const char * *fileN*, int *lineN*, const char * *func*) [static]

Parameters:

← ***e,:*** EventSource from which the log event will be broadcast

in[ll: logging severity level

← ***msg,:*** the message string

← ***fileN,:*** name of the source file producing the log message

← ***lineN,:*** line number of source file where the log message occurs

Returns:

: 1 if message was sent to log relay system, 0 otherwise

9.460.2.4 void Logger::SendLogMessage (EventSource * *e*, LogLevel *ll*, const char * *msg*, const char * *fileN*, int *lineN*, const char * *func*) [static]

Parameters:

← ***e,:*** EventSource from which the log event will be broadcast

in[ll: logging severity level

← ***msg,:*** the message string

← ***fileN,:*** name of the source file producing the log message

← ***lineN,:*** line number of source file where the log message occurs

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

- het_lib/logger/include/Logger.h
- het_lib/logger/src/Logger.cpp

9.461 LoggerDB Class Reference

Public Member Functions

- sqlite3 ∗ **open** (const char ∗dbFile)
- sqlite3 ∗ **GetHandle** ()
- int **insert** (Event ∗e)
- void **Close** ()

Public Attributes

- const std::string **createStmt**

Protected Member Functions

- bool **QueryTabExists** (sqlite3 ∗db, std::string &stmtStr)
- bool **QueryTabExists** (sqlite3 ∗db)
- bool **createLogTb** (sqlite3 ∗db)
- bool **AttribTabInsert** (sqlite3_int64 event_id, std::string &key, std::string &value, const std::string &data_type_string)
- bool **EventTabInsert** (sqlite3_int64 &event_id, fptime msgTime, const std::string &system, const std::string &source, const std::string &key)

Protected Attributes

- bool **tbExists_**

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

- het_lib/logger/include/loggerdb.h
- het_lib/logger/src/loggerdb.cpp

9.462 LogRelayConfig Class Reference

Public Member Functions

- **LogRelayConfig** (bool isNamed=false)

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

- logrelay/tcs_log_relay_config.h

9.463 LogRelayDriver Class Reference

Public Member Functions

- **LogRelayDriver** (const string &appName, LogRelayConfig *config)
- int **start** ()
- void **stop** ()
- void process (Message *m)

9.463.1 Member Function Documentation

9.463.1.1 void LogRelayDriver::process (Message * *message*) [virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- logrelay/tcs_log_relay_driver.h
- logrelay/tcs_log_relay_driver.cpp

9.464 LogServerException Class Reference

Public Member Functions

- **LogServerException** (tcsexcept::tcs_exception_t __logserver, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.465 HETillumination::M1motion Struct Reference

Public Attributes

- float **dx2dtheta_itf**
- float **dx2dphi_itf**
- float **dy2dtheta_itf**
- float **dy2dphi_itf**
- float **drho2drho_itf**
- float **dx2dtel_x**
- float **dx2dtel_y**
- float **dy2dtel_x**
- float **dy2dtel_y**
- float **drho2dcarriage**
- float **drho2darm**

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

- het_lib/illum_lib/include/illumination.h

9.466 HETillumination::M4motion Struct Reference

Public Attributes

- float **dx2dtheta_itf**
- float **dx2dphi_itf**
- float **dy2dtheta_itf**
- float **dy2dphi_itf**
- float **drho2drho_itf**
- float **dx2dtel_x**
- float **dx2dtel_y**
- float **dy2dtel_x**
- float **dy2dtel_y**
- float **drho2dcarriage**
- float **drho2darm**

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

- het_lib/illum_lib/include/illumination.h

9.467 m7302::M7302 Class Reference

Public Member Functions

- def **__init__**
- def **connect**
- def **disconnect**
- def **command**
- def **set**
- def **status**
- def **version**

Public Attributes

- **name**
- **port**
- **host**
- **log**
- **csock**
- **rfile**
- **wfile**
- **lock**
- **connected**

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

- integration/m7302.py

9.468 mainWindow Class Reference

Classes

- class DashBoardAlerts

Public Types

- enum { **WARNINGS_TAB** = 0, **ERRORS_TAB**, **FATAL_ERRORS_TAB**, **ALARMS_TAB** }

Public Slots

- void **CommandIssued** (QString, QString)
- void **CommandDone** (QString, bool)
- void **CommandError** (QString, QString)
- void **CommandIssued** (QString, QString)
- void **CommandDone** (QString, bool)
- void **CommandError** (QString, QString)

Signals

- void **ForceLegacyStatusEvents** (void)
- void **AbortDome** (void)
- void **AbortShutter** (void)
- void **StopStructure** (void)
- void **AbortTracker** (void)
- void **PlaySound** (QString, bool)
- void **StopSound** (QString)
- void **TrajectoryStartTime** (double, QString)
- void **TrajectoryRemainingTime** (double, QString)

Public Member Functions

- **mainWindow** (guiConfig *pConfig, QWidget *pParent=NULL)
- virtual void **closeEvent** (QCloseEvent *)
- void **printDetachedTabs** (void)
- void **AttachTab** (QWidget *pParent)
- void **playWarningSound** (void)
- void **playErrorSound** (void)
- void **playFatalErrorSound** (void)
- void **playAlarmSound** (void)
- **mainWindow** (guiConfig *pConfig, QWidget *pParent=NULL)
- virtual void **closeEvent** (QCloseEvent *)
- void **printDetachedTabs** (void)
- void **AttachTab** (QWidget *pParent)

Protected Member Functions

- void **updateTrajectoryStartTime** (void)
- void **updateDashBoard** (void)
- void **updateDashBoardTimes** (void)
- void **updateDashBoardStatus** (void)
- void **updateDashBoardSubSystemAlerts** (void)
- void **updateDashBoardAlerts** (const string &subsys, DashBoardAlerts &dashBoardAlerts, bool running, QPushButton *pPushButton)
- void **startCommonDashBoardAlerts** (const string &subsystem, bool running, DashBoardAlerts &dashBoardAlerts)
- void **finishCommonDashBoardAlerts** (bool running, DashBoardAlerts &dashBoardAlerts, QPushButton *pPushButton)
- void **ClearAlerts** (int tabIndex)
- void **updateTrackerDashBoardAlerts** (DashBoardAlerts &dashBoardAlerts)
- void **updateStructureDashBoardAlerts** (DashBoardAlerts &dashBoardAlerts)
- void **updateLegacyDashBoardAlerts** (DashBoardAlerts &dashBoardAlerts)
- void **updateDomeDashBoardAlerts** (DashBoardAlerts &dashBoardAlerts)
- void **updateShutterDashBoardAlerts** (DashBoardAlerts &dashBoardAlerts)
- void **updateWeatherDashBoardAlerts** (DashBoardAlerts &dashBoardAlerts)
- void **updateOCDDashBoardAlerts** (DashBoardAlerts &dashBoardAlerts)
- void **updateIglooDashBoardAlerts** (DashBoardAlerts &dashBoardAlerts)
- QString **setAlertColor** (bool running, bool bWarn, bool bError, bool bFatalError, bool bAlarm)
- QString **addTimestamp** (const char *msg)
- QString **addTimestampAndTerminator** (const char *msg)
- QWidget * **loadUiFile** (QString form)
- void **loadSettings** (void)
- void **saveSettings** (void)
- void **constructTabs** (const QStringList &tabNames)
- void **detachTabs** (void)
- void **addToGridLayout** (QWidget *pWidget, const int &nCombinningRow, const int &nCombinningCol, const int &nRowSpan, const int &nColSpan)
- void **enableAlertPushButtons** (void)
- QWidget * **loadUiFile** (QString form)
- void **loadSettings** (void)
- void **saveSettings** (void)
- void **constructTabs** (const QStringList &tabNames)
- void **detachTabs** (void)
- void **addToGridLayout** (QWidget *pWidget, const int &nCombinningRow, const int &nCombinningCol, const int &nRowSpan, const int &nColSpan)

Protected Attributes

- DashBoardAlerts **m_TCSDashBoardAlerts**
- DashBoardAlerts **m_PASDashBoardAlerts**
- DashBoardAlerts **m_PFIPDashBoardAlerts**
- DashBoardAlerts **m_TrackerDashBoardAlerts**
- DashBoardAlerts **m_LegacyDashBoardAlerts**
- DashBoardAlerts **m_StructureDashBoardAlerts**
- DashBoardAlerts **m_DomeDashBoardAlerts**

- DashBoardAlerts **m_ShutterDashBoardAlerts**
- DashBoardAlerts **m_WeatherDashBoardAlerts**
- DashBoardAlerts **m_SAMSDashBoardAlerts**
- DashBoardAlerts **m_APCDashBoardAlerts**
- DashBoardAlerts **m_CCASDashBoardAlerts**
- DashBoardAlerts **m_VIRUSDashBoardAlerts**
- DashBoardAlerts **m_LRS2DashBoardAlerts**
- DashBoardAlerts **m_HRSDashBoardAlerts**
- DashBoardAlerts **m_HPFDashBoardAlerts**
- DashBoardAlerts **m_OCDDashBoardAlerts**
- DashBoardAlerts **m_NamedDashBoardAlerts**
- DashBoardAlerts **m_LogRelayDashBoardAlerts**
- DashBoardAlerts **m_EventMonitorDashBoardAlerts**
- DashBoardAlerts **m_IglooDashBoardAlerts**
- guiConfig * **m_pConfig**
- QTimer * **m_pDashBoardTimesTimer**
- QTimer * **m_pDashBoardStatusTimer**
- QLabel * **m_pPositionIndicator**
- double **m_UntilTrajectoryStarts_secs**
- QShortcut * **m_pToggleMenuBarShortcut**
- QShortcut * **m_pDetachTabShortcut**
- QShortcut * **m_pExitShortcut**
- bool **m_bSilenceWarnings**
- bool **m_bSilenceErrors**
- bool **m_bSilenceFatalErrors**
- bool **m_bSilenceAlarms**
- QStringList **m_tabNames**
- QList< loadedUi * > **m_loadedWidgets**
- QList< QWidget * > **m_combinedTabs**
- QList< QGridLayout * > **m_gridLayouts**
- QList< detachedTabWindow * > **m_detachedTabs**
- std::map< int, int > **m_gridMaxColWidth**
- std::map< int, int > **m_gridMaxRowHeight**
- std::map< int, int > **m_rowStretch**
- std::map< int, int > **m_colStretch**
- QMap< QString, int > **m_commandStatusMap**

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

- gui/include/mainWindow.h
- utility_gui/include/mainWindow.h
- gui/build/moc_mainWindow.cpp
- gui/src/mainWindow.cpp
- utility_gui/src/mainWindow.cpp

9.469 Ui::mainWindow Class Reference

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

- gui/build/ui_mainWindow.h

9.470 TCSConfig::mapEntry Struct Reference

Public Attributes

- lookupType_t **type**
- void * **pValue**

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

- het_lib/common/include/tcs_config.h

9.471 MAXONStatusInfo Class Reference

Public Member Functions

- void **SetDriveStatus** (MAXONDriveStatus &s)
- void **GetDriveStatus** (MAXONDriveStatus &s)
- void **SetDriveData** (MAXONDriveData &d)
- void **GetDriveData** (MAXONDriveData &d)
- bool **IsDisabledEither** ()
- bool **IsEnabledBoth** ()
- bool **ProbeOrArmIsInError** ()
- bool **ProbeOrArmIsInErrorOrWarning** ()
- bool **TargetReached** (string probeName)
- bool **TargetReachedEither** ()
- bool **IsError** ()
- bool **IsEnabled** ()
- void **LogErrors** ()
- void **SetEnabled** ()

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

- pfip/include/MAXONStatus.h
- pfip/src/PFIPStatus.cpp

9.472 mbintf_bitfield_t Struct Reference

9.472.1 Detailed Description

Structure describing a single register bit, holding a boolean value.

`#include <hwdatadefs.h>`

Public Attributes

- mbreg_bit_t bit
- const char * label
- bool normal
- mbintf_alrtlvl_t alrtlvl
- const char * desc

9.472.2 Member Data Documentation

9.472.2.1 mbintf_alrtlvl_t mbintf_bitfield_t::alrtlvl

Severity level for abnormal values

9.472.2.2 mbreg_bit_t mbintf_bitfield_t::bit

Position within the 16-bit register (0-15).

9.472.2.3 const char* mbintf_bitfield_t::desc

UI description of this bit's purpose

9.472.2.4 const char* mbintf_bitfield_t::label

Short label identifying this bit value

9.472.2.5 bool mbintf_bitfield_t::normal

The expected value to be expected under normal (non-alert) conditions

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

- igloo_sensors/hwdatadefs.h

9.473 mbintf_hldreg_bitfield_t Struct Reference

9.473.1 Detailed Description

Structure describing a read/write register containing up to 16 individual bit values.

`#include <hwdatadefs.h>`

Public Attributes

- mbintf_hldregid_t reg_id
- mbintf_inpregid_t mreg_id
- const char * label
- uint8_t bitfields_ct
- const mbintf_bitfield_t * bitfields

9.473.2 Member Data Documentation

9.473.2.1 const mbintf_bitfield_t* mbintf_hldreg_bitfield_t::bitfields

Array of bit description structures

9.473.2.2 uint8_t mbintf_hldreg_bitfield_t::bitfields_ct

Number of identified bits in this register.

9.473.2.3 const char* mbintf_hldreg_bitfield_t::label

Short label for this register.

9.473.2.4 mbintf_inpregid_t mbintf_hldreg_bitfield_t::mreg_id

Modbus identifier of the corresponding read-only "mirror" register, for validating that values have been written and processed.

9.473.2.5 mbintf_hldregid_t mbintf_hldreg_bitfield_t::reg_id

Modbus identifier of the register.

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

- igloo_sensors/hwdatadefs.h

9.474 mbintf_inpreg_analog_t Struct Reference

9.474.1 Detailed Description

Structure describing a read-only analog register, and its intended user interpretation.

`#include <hwdatadefs.h>`

Public Attributes

- mbintf_inpregid_t reg_id
- const char * label
- double limit_lower
- double limit_upper
- double regval_mult
- double regval_div
- mbintf_alrtlvl_t alrtlvl
- const char * desc

9.474.2 Member Data Documentation

9.474.2.1 mbintf_alrtlvl_t mbintf_inpreg_analog_t::alrtlvl

Severity level for abnormal (out of range) values

9.474.2.2 const char* mbintf_inpreg_analog_t::desc

UI description of this register's purpose

9.474.2.3 const char* mbintf_inpreg_analog_t::label

Short label for this register.

9.474.2.4 double mbintf_inpreg_analog_t::limit_lower

Lowest valid/expected interpreted value.

9.474.2.5 double mbintf_inpreg_analog_t::limit_upper

Highest valid/expected interpreted value.

9.474.2.6 mbintf_inpregid_t mbintf_inpreg_analog_t::reg_id

Modbus identifier of the register.

9.474.2.7 double mbintf_inpreg_analog_t::regval_div

Divisor to apply to the raw register value, to convert to interpreted value.

9.474.2.8 double mbintf_inpreg_analog_t::regval_mult

Multiplier to apply to the raw register value, to convert to interpreted value.

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

- igloo_sensors/hwdatadefs.h

9.475 mbintf_inpreg_bitfield_t Struct Reference

9.475.1 Detailed Description

Structure describing a read-only register containing up to 16 individual bit values.

`#include <hwdatadefs.h>`

Public Attributes

- mbintf_inpregid_t reg_id
- const char * label
- uint8_t bitfields_ct
- const mbintf_bitfield_t * bitfields

9.475.2 Member Data Documentation

9.475.2.1 const mbintf_bitfield_t* mbintf_inpreg_bitfield_t::bitfields

Array of bit description structures.

9.475.2.2 uint8_t mbintf_inpreg_bitfield_t::bitfields_ct

Number of identified bits in this register.

9.475.2.3 const char* mbintf_inpreg_bitfield_t::label

Short label of this register.

9.475.2.4 mbintf_inpregid_t mbintf_inpreg_bitfield_t::reg_id

Modbus identifier of the register.

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

- igloo_sensors/hwdatadefs.h

9.476 mbreg_ptr_t Union Reference

9.476.1 Detailed Description

Union to contain a pointer to a description structure for read-only or read/write binary register, or a read-only analog register value.

`#include <hwdatadefs.h>`

Public Member Functions

- **mbreg_ptr_t** (const mbintf_hldreg_bitfield_t *p_reg)
- **mbreg_ptr_t** (const mbintf_inpreg_bitfield_t *p_reg)
- **mbreg_ptr_t** (const mbintf_inpreg_analog_t *p_reg)

Public Attributes

- const mbintf_hldreg_bitfield_t * **p_hld_bitreg**
- const mbintf_inpreg_bitfield_t * **p_inp_bitreg**
- const mbintf_inpreg_analog_t * **p_inp_anlgreg**

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

- igloo_sensors/hwdatadefs.h

9.477 mbreg_t Struct Reference

9.477.1 Detailed Description

Structure abstracting a register value from a user's perspective. Instances of this structure describe either a single read-only or read/write bit, or a read-only analog value. Constructors are provided based on pointers to description structures, and strings provided by user messages, providing a bidirectional conversion between these strings and the details of the modbus register.

`#include <hwdatadefs.h>`

Public Member Functions

- **mbreg_t** (bool is_hld, int bit_id)
- **mbreg_t** (const mbintf_hldreg_bitfield_t *p_reg, bool is_hld, int bit_id)
- **mbreg_t** (const mbintf_inpreg_bitfield_t *p_reg, bool is_hld, int bit_id)
- **mbreg_t** (const mbintf_inpreg_analog_t *p_reg, bool is_hld, int bit_id)
- **mbreg_t** (std::string label)
- std::string get_name (void)

Public Attributes

- mbreg_ptr_t m_regptr
- bool m_is_hld
- int m_bit_id

9.477.2 Member Function Documentation

9.477.2.1 std::string mbreg_t::get_name (void)

Return string describing the instance, which can be used in messages to get or set the register value.

9.477.3 Member Data Documentation

9.477.3.1 int mbreg_t::m_bit_id

Index of the bitfield (NOT the bit position), or -1 for an analog register (ie. all 16 bits)

9.477.3.2 bool mbreg_t::m_is_hld

true if this is a holding register, false if an input register

9.477.3.3 mbreg_ptr_t mbreg_t::m_regptr

A pointer to the data structure containing the characteristics of this register value

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

- igloo_sensors/hwdatadefs.h

- igloo_sensors/hwdatadefs.cpp

9.478 mbregid_t Union Reference

9.478.1 Detailed Description

Union to contain a 16-bit read-only or read/write register identifier. The intended type of the contents is to be determined externally.

`#include <hwdatadefs.h>`

Public Member Functions

- **mbregid_t** (int reg)
- **mbregid_t** (mbintf_hldregid_t hld)
- **mbregid_t** (mbintf_inpregid_t inp)

Public Attributes

- mbintf_hldregid_t **hldregid**
- mbintf_inpregid_t **inpregid**

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

- igloo_sensors/hwdatadefs.h

9.479 MBRegVal Class Reference

Message handling abstraction providing get/set access to register values, via the modbus communication thread IglooSensorsConnection::comm_thread().

9.479.1 Detailed Description

Instances of this class represent individually addressable values of registers for the purpose of reporting state and allowing user or automated control; this abstraction is provided by the mbreg_t type, an instance of which this class encapsulates. MBRegVal provides an interface to access and modify register values by means of a socket connection to a worker thread, which interleaves these get/set operations with regular read operations for UI and archival events reporting the full state of the PLC.

`#include <hwcomm.h>`

Public Member Functions

- **MBRegVal** (mbintf_hldregid_t regid, int bit_id)
- **MBRegVal** (mbintf_inpregid_t regid, int bit_id)
- **MBRegVal** (std::string label)
- std::string **get_name** (void)
- void **set_value** (zsock_t *zsk_cmd, bool val)
- void **ReplyValue** (zsock_t *zsk_cmd, Message &reply)

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

- igloo_sensors/hwcomm.h
- igloo_sensors/hwcomm.cpp

9.480 md5::MD5 Class Reference

Classes

- struct **__context_t**

Public Member Functions

- void **Init** ()
- void **Update** (const unsigned char *input, unsigned int inputLen)
- void **Final** ()
- void writeToString ()

  Buffer must be 32+1 (nul) = 33 chars long at least.

- char * digestFile (const char *filename)

  Load a file from disk and digest it.

- char * digestMemory (const BYTE *memchunk, int len)

  Digests a byte-array already in memory.

- char * **digestString** (const char *string)

Public Attributes

- BYTE **digestRaw** [16]
- char **digestChars** [33]

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

- camra/md5.h

9.481 Message Class Reference

9.481.1 Detailed Description

Message class encapsulating a received message passed to the Handler::process() remote method and returned by the Sender::send() methods as a response. The class allows application code to set/get parameters and/or data, treating the message payload as one or both of a set of key value pairs and/or data buffer.

`#include <message.h>`

Public Member Functions

- **Message** (const std::string &contents)
- **Message** (const void *data, size_t size)
- **Message** (Payload &p)
- **Message** (Event &e)
- sigjmp_buf * interrupt_mark (void)
- msgid_t id (void)
- void respond (Message *response, bool status=true)
- void done (Message *response=NULL)
- void **done** (TCSBaseException &e)
- void **done** (std::exception &e, tcsexcept::tcs_exception_t t=tcsexcept::__unknown, const std::string &file="unknown", int line=0, const std::string &func="unknown")
- void interrupt (void)
- bool is_error (void)
- bool is_done (void)
- fptime get_receive_time (void)
- std::string get_origin (void)

Protected Attributes

- tthread::mutex **message_mtx_**

Friends

- class Payload
- class InterruptHandler
- class HelpHandler
- class StatusHandler
- class TapHandler
- class UptimeHandler
- class Receiver
- class Sender
- class BusyHandler
- class UnhandledHandler
- int **receiver_recv_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **receiver_run_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **sender_ack_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **sender_done_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **sender_recv_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)

9.481.2 Member Function Documentation

9.481.2.1 void Message::done (Message * *response* = NULL)

Method that can be called to notify the remote side that the caller is done with the Message. Upon return, further communication with the remote side is no longer available. If a response is given, it is deallocated prior to returning. If the message corresponding to this Message was sent synchronously, then the response passed to Message::done() will be returned to the caller on the remote end. Otherwise, it will be pushed into the Sender's message queue.

9.481.2.2 string Message::get_origin (void)

Returns the origin for messages

9.481.2.3 fptime Message::get_receive_time (void)

Returns the receive time for messages that were received. Returns a time of zero for all others.

9.481.2.4 msgid_t Message::id (void)

Returns the identifier for the Message.

9.481.2.5 void Message::interrupt (void)

Sends a SIGUSR1 to the thread in which the Message is handled. This will trigger a jump to the scope of a MESSAGE_SETJMP invocation within the MessageHandler::process() forcing a HandlerInterrupt exception. This allows a running MessageHandler thread to be cancelled. Exception handling within the MessageHandler::process() should be implemented in such a way as to ensure that the method exits cleanly upon interrupt.

9.481.2.6 sigjmp_buf * Message::interrupt_mark (void)

Returns a pointer to the `sigjmp_buf` array associated with this message. This method is public to support the MESSAGE_SETJMP macro and it is assumed that use of this array external to the library is limited to the scope of the macro.

9.481.2.7 bool Message::is_done (void)

Returns true if the Message was flagged as a "done" response by the sender. This is only defined for response Messages.

9.481.2.8 bool Message::is_error (void)

Returns true if the Message was flagged as an error by the sender. This is only defined for response Messages.

9.481.2.9 void Message::respond (Message * *response*, bool *status* = `true`)

Interface for responding to the Message with a given `response` Message. Ownership of the `response` given is assumed by the call and the Message is deallocated prior to the return. Any identifier set for the `response` Message is overwritten by that of the invoking Message. If `status` is false, then response is flagged as an error. For Messages that only emit a single response, the Message::done() method should be used rather than Message::respond() to ensure that the remote caller is returned the response message rather than it being pushed into the Sender's message queue.

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

- het_lib/messaging/include/message.h
- het_lib/messaging/src/message.cpp

9.482 MessageHandler Class Reference

9.482.1 Detailed Description

The Handler class provides an entry point through which the receiving end of a messaging instance may expose an object within the scope of the remote Sender.

`#include <message_handler.h>`

Public Member Functions

- **MessageHandler** (const MessageHandlerDescriptor &d)
- virtual void process (Message *message)=0

Friends

- class Receiver
- int **sender_recv_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **receiver_recv_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **receiver_run_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)

9.482.2 Member Function Documentation

9.482.2.1 virtual void MessageHandler::process (Message * *message*) [pure virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implemented in camra_handlers::reset, camra_handlers::turn_off_leds, camra_handlers::set_ccd_temp, camra_handlers::set_bias_offset, camra_handlers::get_ccd_temp, camra_handlers::get_cryo_temp, camra_handlers::get_cryo_pressure, camra_handlers::get_heater_voltage, camra_handlers::read_now, camra_handlers::abort, camra_handlers::expose, camra_handlers::open_shutter, camra_handlers::close_shutter, camra_handlers::enable_shutter, camra_handlers::disable_shutter, camra_handlers::get_exposure_info, camra_handlers::get_hardware_status, camra_handlers::enable_pressure_gauges, camra_handlers::disable_pressure_gauges, camra_handlers::wait_for_shutter_close, camra_handlers::wait_for_readout, camra_handlers::wait_for_write, camra_handlers::set_clock_voltages, camra_handlers::reset_clock_voltages, camra_handlers::get_clock_voltages, camra_handlers::command_controller, camra_handlers::reset_mux, camra_handlers::reset_tcs, camra_handlers::populate_onewire_device_strings, camra_handlers::set_ccd_temp_xfer, camra_handlers::set_cryo_temp_xfer, camra_handlers::override_ifu_associations, SubSystemHandler, ServerHandler, HelpHandler, InterruptHandler, StatusHandler, TapHandler, UptimeHandler, BusyHandler, UnhandledHandler, GenerateExceptionHandler, NOOPHandler, TimeoutHandler, test, set, get, check, get, set, toggle, DomeStatusHandlerClass, ForceStatusEventsHandlerClass, ShutterSetPowerHandlerClass, ShutterStatusHandlerClass, StructureOnLineHandlerClass, StructureMoveToTrueAzHandlerClass, StructureMoveToStructAzHandlerClass, StructureMoveToCCASHandlerClass, StructureStatusHandlerClass, StructureStowHandlerClass,

WeatherOnLineHandlerClass, WeatherGetDataHandlerClass, LogRelayDriver, MonitorDriver, DevicePipelineSetGROCHandlerClass, DevicePipelineStatusHandlerClass, DevicePipelineEnableSACDMIHandlerClass, DevicePipelineEnableCCASDMIHandlerClass, ImagePipelineStartSendingImagesHandlerClass, ImagePipelineGetRegionOfInterestHandlerClass, ImagePipelineGetExposureTimeHandlerClass, ImagePipelineSetOverScanHandlerClass, ImagePipelineGetOverScanHandlerClass, ImagePipelineSetFilterExposureTimeFactorHandlerClass, ImagePipelineSetFilterSequenceHandlerClass, ImagePipelineSetFilterZeroPointHandlerClass, ImagePipelineGetGainHandlerClass, ImagePipelineSetObjectAndMagnitudesHandlerClass, ImagePipelineSetCleanImagesHandlerClass, ImagePipelineSetCleaningBiasTypeHandlerClass, ImagePipelineSetProcessingTypeHandlerClass, ImagePipelineGetPhotPhotometryOptionsHandlerClass, ImagePipelineGetWFSPhotometryOptionsHandlerClass, ImagePipelineSetPhotometryOptionsHandlerClass, ImagePipelineSetPhotPhotometryOptionsHandlerClass, ImagePipelineSetWFSPhotometryOptionsHandlerClass, PipelineSetCycleTimeHandlerClass, PipelineSetIntervalSpeedHandlerClass, PipelineStartPipelineHandlerClass, PipelineTakeControlHandlerClass, PipelineRelinquishControlHandlerClass, PipelineCyclePipelineHandlerClass, PFIPHandler, SampleHandlerClass, P, PlotterDriver, ForceStatusEventsHandlerClass, ProxyHandler, tune< T >, get_tuning< T >, add_correction, get_correction, reset, start, stop, activate, deactivate, force_state_event, set_fiducial, offset_fiducial, clear_fiducial, get_fiducial, set_analysis_region, get_analysis_region, inject_metrology, DMI_get_status, set_tiptilt_window_length, center, get_tiptilt_window_length, set_focus_window_length, get_focus_window_length, set_tiptilt_gain, get_tiptilt_gain, set_focus_gain, get_focus_gain, unset_position, get_status, set_position, offset_probe, set_filter, load_trajectory, query_trajectory, correct_for_structure, offset_trajectory, tracker_move_tsf, tracker_move_angular, toggle_gui_token, tracker_abort, tracker_init, tracker_move_itf, tracker_position, go_next, cancel_trajectory, get_current_trajectory_info, get_next_trajectory_info, get_current_local_sidereal_time, get_current_pointing, pfip_open_shutter, pfip_close_shutter, pfip_move_probes, pfip_clear_positions, pfip_insert_acquisition_pickoff, pfip_withdraw_acquisition_pickoff, offset_itf, offset_sky, reset_itf_offset, reset_sky_offset, get_itf_offset, get_sky_offset, get_focus_offset, offset_focus, reset_focus_offset, offset_tiptilt, reset_tiptilt_offset, get_tiptilt_offset, get_radec_offset, reset_radec_offset, set_focus, clear_trajectories, update_setup, wait_for_setup, set_metadata, get_metadata, enable_rho, disable_rho, namedHandler, tracker_handlers::initTMCS, tracker_handlers::setTraj, tracker_handlers::abortMove, tracker_handlers::setBufferSize, tracker_handlers::clearError, tracker_handlers::moveRTF, tracker_handlers::moveTSF, and tracker_handlers::timeToIntercept.

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

- het_lib/messaging/include/message_handler.h
- het_lib/messaging/src/message_handler.cpp

9.483 MessageHandlerDescriptor Class Reference

9.483.1 Detailed Description

Descriptor class providing runtime identification of a MessageHandler instance along with the MessageHandler parameter list.

`#include <message_handler.h>`

Classes

- struct **parm**

Public Member Functions

- **MessageHandlerDescriptor** (const std::string &name, const std::string &system, const std::string &usage, const std::string &version, bool reentrant)
- **MessageHandlerDescriptor** (const MessageHandlerDescriptor &d)
- void **add_required_parm** (const std::string &name, const std::string &type, const std::string &usage)
- void **add_optional_parm** (const std::string &name, const std::string &type, const std::string &usage)
- const std::string & **get_name** (void) const
- const std::string & **get_system** (void) const
- const std::string & **get_usage_header** (void) const
- const std::string & **get_version** (void) const
- const std::string & **get_usage** (void) const
- bool **is_reentrant** (void) const

Public Attributes

- std::string **name_**
- std::string **usage_header_**

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

- het_lib/messaging/include/message_handler.h
- het_lib/messaging/src/message_handler.cpp

9.484 MessageHeader Class Reference

Public Member Functions

- std::string **handler** (void)

Friends

- class Message
- class Tap
- class BusyHandler
- class UnhandledHandler
- class Receiver
- class Sender
- int **receiver_recv_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **receiver_run_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **sender_done_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **sender_recv_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)

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

- het_lib/messaging/include/message.h
- het_lib/messaging/src/message.cpp

9.485 MessagingException Class Reference

Public Member Functions

- **MessagingException** (tcsexcept::tcs_exception_t __messaging, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.486 metadata_entry Struct Reference

Public Attributes

- std::string **key**
- std::string **value**
- std::string **type**

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

- camra/tcs_client.h

9.487 MetaPayload Class Reference

Public Member Functions

- void **set_data_time** (const fptime &timestamp)
- void **set_wire_time** (const fptime &timestamp)
- fptime **get_data_time** (void)
- fptime **get_wire_time** (void)

Protected Member Functions

- **MetaPayload** (const MetaPayload &m)
- void **apply_meta_data** (Payload &p)
- void **deserialize** (Payload &p)

Friends

- class Event
- class Message

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

- het_lib/messaging/include/payload.h
- het_lib/messaging/src/payload.cpp

9.488 Metrology Class Reference

Public Member Functions

- **Metrology** (const TCSServerConfig &config, PFIPClient &pfip, TCSDriver *driver=NULL)
- void **start** (void)
- void **start_default_loops** (void)
- void **stop** (void)
- void **clear** (void)
- void **add_correction** (const Correction &correction)
- Correction **get_correction** (double wait=0)
- bool **active** ()
- void **set_focus** (const double &w)
- DMILoop & **dmi** (void)
- WFSLoop & **wfs1** (void)
- WFSLoop & **wfs2** (void)
- WFSLoop & **cwfs** (void)
- GuideLoop & **tt** (void)
- PFIPClient & **pfip** (void)

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

- tcs/metrology.h
- tcs/metrology.cpp

9.489 MetrologyHandler Class Reference

Public Member Functions

- **MetrologyHandler** (MetrologyLoop *l, const MessageHandlerDescriptor &m)

Public Attributes

- MetrologyLoop * **metrology_**

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

- tcs/metrology.h
- tcs/metrology.cpp

9.490 MetrologyLoop Class Reference

Public Member Functions

- **MetrologyLoop** (Instrument &i, const TCSServerConfig &config, TCSDriver *driver, Metrology *master=NULL)
- virtual void **start** (Message &m)
- virtual void **start** (void)
- virtual void **stop** (bool have_lock=0)
- virtual void **clear** (bool have_lock=0)
- virtual void **activate** (void)
- virtual void **deactivate** (void)
- virtual void **force_state_event** (void)
- bool **enabled** (void)
- bool **active** (void)
- Correction **get_correction** (double wait=0)
- virtual void **add_correction** (const Correction &correction, const fptime &data_time)
- void **add_correction** (Message *message)
- virtual void **apply_offset** (Message *message)
- virtual void **apply_metrology** (Payload &p, const fptime &data_time, bool do_throw=0)
- Instrument & **instrument** (void)

Protected Attributes

- Instrument & **instrument_**
- const TCSServerConfig & **config_**
- tthread::thread * **thread_**
- tthread::mutex **mutex_**
- tthread::condition_variable **cvar_**
- volatile bool **enabled_**
- volatile bool **active_**
- Correction **correction_**
- TCSDriver * **driver_**
- MetrologyLoopID **loop_id_**
- Metrology * **master_**
- PASCameraClient **pas_**

Friends

- class MetrologyHandler

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

- tcs/metrology.h
- tcs/metrology.cpp

9.491 MetrologyPIDLoop< T > Class Template Reference

template<typename T> class MetrologyPIDLoop< T >

Public Member Functions

- **MetrologyPIDLoop** (const std::string &name)
- void **set_tuning** (double kp, double ki, double kd)
- Message * **tuning** (void)

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

- tcs/metrology.h

9.492 MidRangeProbes Class Reference

9.492.1 Detailed Description

Move all the probes and their arms to the middle of their respective ranges. If one or more of the probes is diabled by configuration, that probe will not be affected. The PFIPInterrupt() handler can be used to cancel this operation.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **MidRangeProbes** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.493 MirrorDistance Struct Reference

9.493.1 Detailed Description

Computes the distance to the HET mirror.

Based on on my_f by M.E.Cornell.

--ORIGINAL DOCUMENTATION BEGINS HERE--

my_f - compute the distance from the center of the mirror

06JAN11 Modified by M. E. Cornell from nsotraj.c dated 31DEC10, with ideas from W. Moreira's best.c. 14JAN11 slaMap takes TDB, not UTC.

--ORIGINAL DOCUMENTATION ENDS HERE--

AUG 2013 Integrated into point library by Martin Landriau To be used in conjuction with cure/libmath-/ameoba.

Public Types

- enum { **NPars** = 2 }
- typedef double **value_type**

Public Member Functions

- **MirrorDistance** (Point &p, const weather &cw)
- template<int M>
  double **operator()** (const ltl::FVector< double, 2, M > &x) const

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

- tcs/traj_comp/nsotraj.cpp

9.494 AcqMiscStatus::misc_status Struct Reference

Public Member Functions

- **INT_VAR** (ditherPosition, 1)
- **BOOL_VAR** (sbc1Power)
- **BOOL_VAR** (sbc1PowerAlert)
- **BOOL_VAR** (sbc2Power)
- **BOOL_VAR** (sbc2PowerAlert)
- **BOOL_VAR** (sw1Power)
- **BOOL_VAR** (sw1PowerAlert)
- **BOOL_VAR** (sw2Power)
- **BOOL_VAR** (sw2PowerAlert)
- **BOOL_VAR** (bibCameraPower)
- **BOOL_VAR** (bibCameraPowerAlert)
- **BOOL_VAR** (ttCameraPower)
- **BOOL_VAR** (ttCameraPowerAlert)
- **BOOL_VAR** (hpf_acqCameraPower)
- **BOOL_VAR** (hpf_acqCameraPowerAlert)
- **BOOL_VAR** (hr_sasCameraPower)
- **BOOL_VAR** (hr_sasCameraPowerAlert)

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

- gui/include/pfipStatusData.h

9.495 MiscInfo Class Reference

Public Member Functions

- void **SendMiscInfo** ()
- bool **GetBIBCameraPower** ()
- bool **GetTTCameraPower** ()
- bool **GetHPF_ACQCameraPower** ()
- bool **GetHR_SASCameraPower** ()
- bool **GetSBC1Power** ()
- bool **GetSBC2Power** ()
- bool **GetSw1Power** ()
- bool **GetSw2Power** ()
- int **GetDither** ()
- void **SetBIBCameraPower** (bool power)
- void **SetTTCameraPower** (bool power)
- void **SetHPF_ACQCameraPower** (bool power)
- void **SetHR_SASCameraPower** (bool power)
- void **SetSBC1Power** (bool power)
- void **SetSBC2Power** (bool power)
- void **SetSw1Power** (bool power)
- void **SetSw2Power** (bool power)
- void **SetcRIOErrors** (string errors)
- void **SetcRIOIterationDuration** (int mainLoopDuration, int shutterLoopDuration)
- void **SetDither** (pfip::DitherPos val)

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

- pfip/include/PFIPStatus.h
- pfip/src/EventInfo.cpp

9.496 ModalFitData Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual void **newData** (int size)
- virtual void **deleteData** (void)

Public Attributes

- ushort **Nmode**
- double * **c**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.497 ModalMatrix Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual void **newData** (int size1, int size2)
- virtual void **deleteData** (void)

Public Attributes

- ushort **width**
- ushort **height**
- ushort **maxterm**
- gsl_matrix * **modal**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.498 modbus_mapping_t Struct Reference

Public Attributes

- int **nb_bits**
- int **nb_input_bits**
- int **nb_input_registers**
- int **nb_registers**
- uint8_t * **tab_bits**
- uint8_t * **tab_input_bits**
- uint16_t * **tab_input_registers**
- uint16_t * **tab_registers**

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

- pfip/MODBUS_lib/include/modbus.h

9.499 MODBUSComm Class Reference

9.499.1 Detailed Description

A class to manage a modbus connection. In the MODBUS protocol, the two processes communicate using an array of short unsigned integers called holding registers. The holding registers are identified by their offset in the array. Serialization for call to read and write to the modbus server is provided using a Mutex serialize_modbus. Throws PFIP_MODBUSOffsetsException if there is an error with modbus commmunication The methods for the ModBusComm class allow multiple registers to be set on one modbus call. There is a boolean array containing indicators that a particular value should be written to the modbus server (sendThisVal). The register array (regData) holds the values that are to be sent. To make a modbus write transaction, use lockEPICSAccess, followed by a series of setValues calls to establish the register values, and finally, a sendCommand to push the register values to the server.

`#include <MODBUSComm.h>`

Public Member Functions

- MODBUSComm (const string &sourceName)
  Constructor.
- virtual ~MODBUSComm ()
  Destructor.
- int Connect (const char *ip, const int port, const int maxRegs)
  Connect to a modbus ip/port, provide maximum number of holding registers used by this connection.
- int getValues (const int ValueID, const int numVals, unsigned short *const Value)
  Retrieve a group of holding register values starting with the one identified by ValueID.
- void setValues (const int ValueID, const int numValues, unsigned short *const Values)
  Set values for one or more contiguous registers to be sent starting with the one identified by ValueID.
- void lockMODBUSAccess ()
  Initialize a series of updates to the data array, zeroes history of setValues calls Call this before calling a series of setValues calls.
- void **unlockMODBUSAccess** ()
- void sendCommand ()
  Send a series of register values to the MODBUS server, previously set by call to setValues.
- void close ()
  End the connection.

9.499.2 Member Function Documentation

9.499.2.1 int MODBUSComm::Connect (const char * *ip*, const int *port*, const int *maxRegs*)

Parameters:

← ***ip*** - a string containing the ip address (ex. "192.162.1.222")

← ***port*** - the port number that MODBUS will use (ex. 502)

← ***maxRegs*** - number of holding registers (individual data values) used in this modbus connection

9.499.2.2 int MODBUSComm::getValues (const int *ValueID*, const int *numVals*, unsigned short *const *Value*)

Parameters:

← ***ValueID*** - the identifier for the modbus register to retrieve

← ***numVals*** - the number of values to retrieve starting with ValueID

→ ***Value*** - a ptr to the array of values (numVals) retrieved

9.499.2.3 void MODBUSComm::setValues (const int *ValueID*, const int *numValues*, unsigned short *const *Values*)

Parameters:

← ***ValueId*** - the identifier for the modbus register to set

← ***numValues*** - number of values to set starting with ValueId

← ***Values*** - ptr to array of Values

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

- pfip/include/MODBUSComm.h
- pfip/src/MODBUSComm.cpp

9.500 MODBUSTime Struct Reference

Public Attributes

- unsigned short millis
  milliseconds
- unsigned short mins
  minutes

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

- pfip/include/MODBUSComm.h

9.501 modified_julian_day Class Reference

Public Member Functions

- **modified_julian_day** (const double &other)

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

- het_lib/common/include/fptime.h

9.502 MonitorConfig Class Reference

Public Member Functions

- **MonitorConfig** (bool isNamed=false)

Public Attributes

- string **system_names** [64]
- string **system_filter**
- string **source_filter**
- string **key_filter**
- string **output_file**
- string **db_file**
- bool **pretty_print**
- bool **log_print**

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

- monitor/tcs_monitor_config.h

9.503 MonitorDriver Class Reference

Public Member Functions

- **MonitorDriver** (const string &appName, MonitorConfig *config)
- int **start** ()
- void **stop** ()
- void process (Message *m)
- void **event_thread** (void)

9.503.1 Member Function Documentation

9.503.1.1 void MonitorDriver::process (Message * *message*) [virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- monitor/tcs_monitor_driver.h
- monitor/tcs_monitor_driver.cpp

9.504 liborchestrate::MonitorSubsystem Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.505 MonPrintRegister Class Reference

Public Member Functions

- **MonPrintRegister** (const string &nm, Receiver *rcv, Alarm_Monitor *mon, MessageHandlerDescriptor &usage)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.506 MonWriteRegister Class Reference

Public Member Functions

- **MonWriteRegister** (const string &nm, Receiver *rcv, Alarm_Monitor *mon, MessageHandlerDescriptor &usage)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

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

- pfip/include/pfipServer.h
- pfip/src/FCUHandlers.cpp

9.507 MountModel Class Reference

Public Member Functions

- **MountModel** (const itf_coordinate &offset, bool identity=false)
- rtf_coordinate itf_to_rtf (itf_coordinate itf, bool include_offset=true)
- itf_coordinate rtf_to_itf (rtf_coordinate rtf, bool include_offset=true)
- itf_coordinate **itf_to_itf** (const itf_coordinate &itf, bool include_offset=true)
- itf_coordinate **tsf_to_itf** (tsf_coordinate)
- tsf_coordinate **itf_to_tsf** (itf_coordinate)
- void rotateXY (itf_coordinate &itf, const double theta, const double phi)
- void **rotateXYinv** (itf_coordinate &itf, const double theta, const double phi)
- void **offset_itf** (const itf_coordinate &offset)
- void **reset_itf_offset** (void)
- itf_coordinate **offset** (void)
- double focus_model_low (const double itfx, const double itfy)
- double **focus_model_high** (const double itfx, const double itfy)
- vec2_t pointing_mount_model_offset (const itf_coordinate &current_tracker, const double azimuth_rad, TrackDir direction)

Static Public Member Functions

- static std::string **get_zero_offset_string** (void)

Friends

- class Point

9.507.1 Member Function Documentation

9.507.1.1 double MountModel::focus_model_low (const double *itfx*, const double *itfy*)

focus mount model broken up into low-order and high-order terms

9.507.1.2 rtf_coordinate MountModel::itf_to_rtf (itf_coordinate *itf*, bool *include_offset* = `true`)

Transform between ITF and RTF. Based on measurements of commanded positions on the tracking sphere.

Transforms ITF x, y, z, theta and phi to RTF x, y, z, theta, phi. Based on measurements of commanded positions on the tracking sphere.

9.507.1.3 vec2_t MountModel::pointing_mount_model_offset (const itf_coordinate & *current_tracker*, const double *azimuth_rad*, TrackDir *direction*)

empirical pointing residual mount model, returns az/el corrections in arcseconds

9.507.1.4 void MountModel::rotateXY (itf_coordinate & *itf*, const double *theta*, const double *phi*)

Make small rotations about X and Y axes; assumes SMALL angles so that the two matrices commute.

9.507.1.5 itf_coordinate MountModel::rtf_to_itf (rtf_coordinate *rtf*, bool *include_offset* = `true`)

Transform from RTF to ITF. The current offset is removed as well.

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

- het_lib/common/include/tcs_coordinates.h
- het_lib/common/src/tcs_coordinates.cpp
- playground/mount_model/itf_to_itf.cpp

9.508 mountModel Class Reference

9.508.1 Detailed Description

Class encapsulating the global mount model

`#include <coords.h>`

Public Member Functions

- **mountModel** (bool applySM, bool applyTTM, bool applyTSM)

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

- het_lib/coord_trans/include/coords.h

9.509 MountModelException Class Reference

Public Member Functions

- **MountModelException** (tcsexcept::tcs_exception_t __mount_model, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.510 MoveProbeParms Class Reference

Public Attributes

- std::string **probeName**
- bool **using_radians**
- double **x**
- double **y**
- bool **x_specified**
- bool **y_specified**
- bool **using_encoder**
- unsigned int **carriage**
- unsigned int **arm**
- bool **carriage_specified**
- bool **arm_specified**

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

- pfip/include/ProbeAssembly.h

9.511 MoveProbes Class Reference

9.511.1 Detailed Description

Move the guider and wave front sensors (wfs) to new coordinates. The coordinates are supplied as x/y in degrees. The x/y coordinates are the point to which the end of the probe arm will move on the focal plane. It is possible to disable the probes from the pfipserver configuration. If any of the probes are disabled, their new coordinate parameters are ignored, and the probe will not move. If there is a fault that prevents the combined probe move from continuing, the moves for all probes will be cancelled. It is possible to cancel a MoveProbes request by using the PFIPInterrupt() handler.

Optional parameters:

guider1x<float> - The x coordinate (in degrees) for the new position of guider probe 1.

Guider1y<float> - The y coordinate (in degrees) for the new position of guider probe 1.

Guider2x<float> - The x coordinate (in degrees) for the new position of guider probe 2.

Guider2y<float> - The y coordinate (in degrees) for the new position of guider probe 2.

WFS1x<float> - The x coordinate (in degrees) for the new position of wave front sensor 1.

WFS1y<float> - The y coordinate (in degrees) for the new position of wave fron sensor 1.

WFS2x<float> - The x coordinate (in degrees) for the new position of wave front sensor 2.

WFS2y<float> - The y coordinate (in degrees) for the new position of wave front sensor 2.

```
#include <pfip_handler_doc.h>
```

Public Member Functions

- **MoveProbes** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.512 MoveProbes_Enc Class Reference

9.512.1 Detailed Description

Move the guiders and wave front sensors (wfs) to new encoder positions. The encoder positions are supplied as carriage and arm for each probe. If there is a fault that prevents the combined probe move from continuing, the moves for all probes will be cancelled. It is possible to cancel a MoveProbes request by using the PFIPInterrupt() handler.

Optional parameters:

Guider1_Carriage<int> - The carriage encoder count requested guider 1.

Guider1_Arm<int> - The arm encoder count requested for guider 1.

Guider2_Carriage<int> - The carriage encoder count requested for guider 2.

Guider2_Arm<int> - The arm encoder count requested for guider 2.

WFS1_Carriage<int> - The carriage encoder count requested for wave front sensor 1.

WFS1_Arm<int> - The arm encoder count requested for wave front sensor 1.

WFS1_Carriage<int> - The carriage encoder count requested wave front sensor 2.

WFS2_Arm<int> - The arm encoder count requested for wave front sensor 1.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **MoveProbes_Enc** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.513 MoveProbes_RaDec Class Reference

9.513.1 Detailed Description

NOTE: This interface is not to be used during normal TCS operations. It is a testing interface only!! Move the guiders and wave front sensors (wfs) to new coordinates. The coordinates are supplied as Ra and Dec for each probe. These values are translated using the other supplied parameters, the focal plane axis Ra and Dec, and the parallactive angle. All angles are assumed to be in degrees. It is possible to disable the probes from the pfipserver configuration. If any of the probes are disabled, their new coordinate parameters are ignored, and the probe will not move. If there is a fault that prevents the combined probe move from continuing, the moves for all probes will be cancelled. It is possible to cancel a MoveProbes request by using the PFIPInterrupt() handler.

Required parameters:

Axis_Ra<float> - The right ascension of the center of the focal plane.

Axis_Dec<float> - The declination of the center of the focal plane.

Pa<float> - The parallactic angle of the target.

Optional parameters:

Guider1_Ra<float> - The right ascension of the requested guider 1 position.

Guider1_Dec<float> - The declination of the requested guider 1 position.

Guider2_Ra<float> - The right ascension of the requested guider 2 position.

Guider2_Dec<float> - The declination of the requested guider 2 position.

WFS1_Ra<float> - The right ascension of the requested position for wave front sensor 1.

WFS1_Dec<float> - The declination of the requested position for wave front sensor 1.

WFS1_Ra<float> - The right ascension of the requested position for wave front sensor 2.

WFS2_Dec<float> - The declination of the requested position for wave front sensor 1.

```
#include <pfip_handler_doc.h>
```

Public Member Functions

- **MoveProbes_RaDec** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.514 moveRTF Class Reference

9.514.1 Detailed Description

Sends a move command in RTF coordinates. Sends with tmcs_mova_rtf or tmcs_movr_rtf command depending on the type parameter.

Required parameters:

type<string> - XXX

vel<string> - XXX

x<double> - XXX

y<double> - XXX

z<double> - XXX

rho<double> - XXX

theta<double> - XXX

phi<double> - XXX

```
#include <tracker_handler_doc.h>
```

Public Attributes

- string **revstring** = r'$Id: moveRTF.py 8153 2017-08-29 19:29:28Z fowler $'
- string **description**
- dictionary pos_dict

  Main action.

- string **pos** = 'none'
- list **pos** = args.vals[0]
- string **velocity** = 'TRK'
- tuple **cli_list** = start_clients(tracker=True)
- list **tracker** = cli_list['tracker']
- tuple **h**

9.514.2 Member Data Documentation

9.514.2.1 string moveRTF::description

Initial value:

```
'''Move to the requested RTF position. The position may be
one of the pre-defined positions or six RTF coordinates.  The default
velocity is track. If both track and slew speeds are set on the
command line, then the script will exit since it does not know what
you want.  Verbose mode will print useful information about what is
happening. Quiet mode will print nothing at all, not even errors.  The
default is to not print the useful information but to print error
messages.

left == (-1800, -1800, +160, 0, 0, 0)

right = (+1800, -1800, +160, 0, 0, 0)
```

```
home = (0, -1800, +160, 0, 0, 0)

zero = (0, 0, 0, 0, 0, 0)'''
```

9.514.2.2 tuple moveRTF::h

Initial value:

```
tracker.moveRTF( type="abs", vel=velocity,
                    x=X_rtf, y=Y_rtf, z=Z_rtf,
                    rho=Rho_rtf, theta=Theta_rtf, phi=Phi_rtf )
```

9.514.2.3 dictionary moveRTF::pos_dict

Initial value:

```
{
    #             X        Y      Z    Rho   Theta   Phi
    'zero' : (    0.0,     0.0,   0.0,   0.0,   0.0,   0.0),
    'home' : (    0.0, -1800.0,   0.0,   0.0,   0.0,   0.0),
    'left' : (-1800.0, -1800.0,   0.0,   0.0,   0.0,   0.0),
    'right': ( 1800.0, -1800.0,   0.0,   0.0,   0.0,   0.0),
    'other': (    0.0,     0.0,   0.0,   0.0,   0.0,   0.0)
}
```

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

- tracker/tracker_handler_doc.h
- scripting/scripts/moveRTF.py

9.515 tracker_handlers::moveRTF Class Reference

Public Member Functions

- **moveRTF** (TrackerDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.515.1 Member Function Documentation

9.515.1.1 void tracker_handlers::moveRTF::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tracker/tracker_handlers.h

9.516 MoveTo Class Reference

Move the dome to the specified azimuth.

9.516.1 Detailed Description

Move the structure to the specified azimuth with mount model applied.

Parameters:

target <double> - target azimuth in degrees

This command will connect to the structure computer, take control, set the command time out, move to the desired position, stop the move, and release control.

Parameters:

position <double> - desired structure azimuth in degrees

```
#include <legacyDomeSubSystem.h>
```

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

- legacy/include/legacyDomeSubSystem.h

9.517 MoveToStructAz Class Reference

Move the structure to the specified azimuth without mount model applied.

9.517.1 Detailed Description

This command will connect to the structure computer, take control, set the command time out, move to the desired position, stop the move, and release control.

Parameters:

position <double> - desired structure azimuth in degrees

```
#include <legacyStructureSubSystem.h>
```

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

- legacy/include/legacyStructureSubSystem.h

9.518 moveTSF Class Reference

9.518.1 Detailed Description

Sends a move command in tracker coordinates.

Required parameters:

type<string> - move type, either rel (relative) or abs (absolute)

vel<string> - velocity string, see ??? (need TMCS reference)

x<double> - x coordinate (mm)

y<double> - y coordinate (mm)

rho<double> - rho coordinate (deg)

h1<double> - Hexapod leg 1 coordinate (mm)

h2<double> - XXX

h3<double> - XXX

h4<double> - XXX

h5<double> - XXX

h6<double> - XXX

```
#include <tracker_handler_doc.h>
```

Public Attributes

- string **revstring** = r'$Id: moveTSF.py 8114 2017-08-18 15:12:25Z fowler $'
- string **description**
- dictionary pos_dict

  Main action.

- string **pos** = 'none'
- list **pos** = args.vals[0]
- string **velocity** = 'TRK'
- tuple **tracker** = start_clients(tracker=True)
- int **count** = 0
- tuple **h**

9.518.2 Member Data Documentation

9.518.2.1 string moveTSF::description

Initial value:

```
'''Move to the requested TSF position. The position may be
one of the pre-defined positions or ten TSF coordinates.  The default
velocity is track. If both track and slew speeds are set on the
command line, then the script will exit since it does not know what
you want.  Verbose mode will print useful information about what is
happening. Quiet mode will print nothing at all, not even errors.  The
default is to not print the useful information but to print error
```

```
messages.

left == (-1800, -1800, 0, 0, 0, 0, 0, 0, 0)

right = (+1800, -1800, 0, 0, 0, 0, 0, 0, 0)

home = (0, -1800, 0, 0, 0, 0, 0, 0, 0)

zero = (0, 0, 0, 0, 0, 0, 0, 0, 0)'''
```

9.518.2.2 tuple moveTSF::h

Initial value:

```
tracker.moveTSF( type="abs",vel=velocity,
                     x=X_tsf,    y=Y_tsf, rho=Rho_tsf,
                     h1=H1_tsf, h2=H2_tsf, h3=H3_tsf,
                     h4=H4_tsf, h5=H5_tsf, h6=H6_tsf )
```

9.518.2.3 dictionary moveTSF::pos_dict

Initial value:

```
{
    #             X         Y    Rho     H1     H2     H3     H4     H5     H6
    'zero' : (    0.0,      0.0,   0.0,   0.0,   0.0,   0.0,   0.0,   0.0,   0.0),

    'home' : (    0.0, -1800.0,   0.0,   0.0,   0.0,   0.0,   0.0,   0.0,   0.0),

    'left' : (-1800.0, -1800.0,   0.0,   0.0,   0.0,   0.0,   0.0,   0.0,   0.0),

    'right': ( 1800.0, -1800.0,   0.0,   0.0,   0.0,   0.0,   0.0,   0.0,   0.0),

    'cmd'  : (    0.0,      0.0,   0.0,   0.0,   0.0,   0.0,   0.0,   0.0,   0.0)
}
```

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

- tracker/tracker_handler_doc.h
- scripting/scripts/moveTSF.py

9.519 tracker_handlers::moveTSF Class Reference

Public Member Functions

- **moveTSF** (TrackerDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.519.1 Member Function Documentation

9.519.1.1 void tracker_handlers::moveTSF::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tracker/tracker_handlers.h

9.520 MsgAttribute Class Reference

Public Attributes

- std::string **keyname**
- std::string **keyvalue**
- sqlite3_int64 **event_id**
- int **data_type**

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

- het_lib/logger/include/loggerdb.h

9.521 tthread::mutex Class Reference

9.521.1 Detailed Description

Mutex class. This is a mutual exclusion object for synchronizing access to shared memory areas for several threads. The mutex is non-recursive (i.e. a program may deadlock if the thread that owns a mutex object calls lock() on that object).

See also:

recursive_mutex

`#include <tinythread.h>`

Public Member Functions

- mutex ()
  Constructor.
- ~mutex ()
  Destructor.
- void lock ()
- bool try_lock ()
- void unlock ()
- **mutex** (const mutex &)
- mutex & **operator=** (const mutex &)

Friends

- class **condition_variable**

9.521.2 Member Function Documentation

9.521.2.1 void tthread::mutex::lock () [inline]

Lock the mutex. The method will block the calling thread until a lock on the mutex can be obtained. The mutex remains locked until `unlock()` is called.

See also:

lock_guard

9.521.2.2 bool tthread::mutex::try_lock () [inline]

Try to lock the mutex. The method will try to lock the mutex. If it fails, the function will return immediately (non-blocking).

Returns:

`true` if the lock was acquired, or `false` if the lock could not be acquired.

9.521.2.3 void tthread::mutex::unlock () [inline]

Unlock the mutex. If any threads are waiting for the lock on this mutex, one of them will be unblocked.

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

- het_lib/tthread/include/tinythread.h

9.522 Mux Class Reference

Class corresponding to the controller multiplexing hardware.

`#include <camra_hardware.h>`

Public Member Functions

- **Mux** (int idx, int id, const CAMRAConfig &config, CAMRAHardware *hw)
- uint16_t * data ()

  Returns a pointer to the start of this Mux's data region.

- map< int, Cntl * > & **cntls** (void)

Protected Attributes

- CAMRAHardware * **camra_**
- map< int, Cntl * > **cntls_**
- uint16_t * **data_**

Friends

- class CAMRAHardware
- class Cntl
- class CCD
- class Amp

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

- camra/camra_hardware.h
- camra/camra_hardware.cpp

9.523 liborchestrate::Named Class Reference

Public Member Functions

- def **__init__**

Public Attributes

- **subsys**

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

- integration/liborchestrate.py

9.524 namedClient Class Reference

Public Member Functions

- std::string **lookup** (const std::string &key)

Static Public Member Functions

- static namedClient & **instance** (const std::string &namedRoute="")

Protected Attributes

- Sender * **m_pSender**

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

- het_lib/common/include/tcs_named.h
- het_lib/common/src/tcs_named.cpp

9.525 namedConfig Class Reference

Public Member Functions

- **namedConfig** (bool isNamed=1)

Public Attributes

- string **m_dbPathname**

Protected Member Functions

- virtual void print_summary (void)

  Called prior to displaying options.

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

9.525.1 Member Function Documentation

9.525.1.1 void namedConfig::print_summary (void) [protected, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.525.1.2 void namedConfig::set_options (OptionParser & *optionParser*) [protected, virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- tcsnamed/tcsnamed.h
- tcsnamed/tcsnamed.cpp

9.526 NamedException Class Reference

Public Member Functions

- **NamedException** (tcsexcept::tcs_exception_t __named, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.527 namedHandler Class Reference

Public Member Functions

- **namedHandler** (namedServer *pServer)
- void process (Message *m)

Protected Attributes

- namedServer * **m_pServer**

9.527.1 Member Function Documentation

9.527.1.1 void namedHandler::process (Message * *message*) [virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcsnamed/tcsnamed.h
- tcsnamed/tcsnamed.cpp

9.528 util::NamedReader Class Reference

Public Member Functions

- **NamedReader** (const string &route, const list< string > optionNameList)
- virtual string **nextOptionName** ()
- virtual string **nextOptionValue** (const Option *op)
- virtual bool **done** ()

Protected Member Functions

- bool **nextNamedEntry** (void)

Protected Attributes

- list< string > **optionNameList_**
- bool **done_**
- list< string >::const_iterator **itr_**
- string **optionname_**
- string **optionvalue_**

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

- het_lib/common/include/named_reader.h
- het_lib/common/src/named_reader.cpp

9.529 namedServer Class Reference

Public Types

- typedef std::map< string, string > **nameDB_t**

Public Member Functions

- **namedServer** (const string &appName, namedConfig *config)
- virtual int **start** (void)
- virtual void **stop** (void)
- bool **init** (void)
- bool **lookup** (const string key, string &value)

Public Attributes

- nameDB_t **m_nameDB**

Protected Member Functions

- void **parseDBFile** (const string &filename)

Protected Attributes

- namedConfig * **m_pConfig**
- Receiver * **m_receiver**

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

- tcsnamed/tcsnamed.h
- tcsnamed/tcsnamed.cpp

9.530 native_wrapper< Type > Class Template Reference

template<typename Type> class native_wrapper< Type >

Public Member Functions

- **native_wrapper** (const Type &other)
- **operator Type** () const

Public Attributes

- Type **value_**

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

- het_lib/common/include/fptime.h

9.531 NoiseAreaData Class Reference

Public Attributes

- bool **flag** [4]
- uint **N**
- float **gross_bias**
- float **gross_ron**
- float **x** [4]
- float **y** [4]
- float **d** [4]
- float **meanval** [4]
- float **stdval** [4]

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

- het_lib/wfs_lib/include/wfslib.h

9.532 noncopyable Class Reference

9.532.1 Detailed Description

Make an object non-copyable (and non construction-copyable) by inheriting from this class which does nothing but declare the copy-constructor and assignment operator private and not implement them.

`#include <noncopyable.h>`

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

- het_lib/common/include/noncopyable.h

9.533 NOOPHandler Class Reference

Public Member Functions

- virtual void process (Message ∗message)

9.533.1 Member Function Documentation

9.533.1.1 virtual void NOOPHandler::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- het_lib/messaging/include/builtin_handlers.h

9.534 tcs_photometry::Object Struct Reference

9.534.1 Detailed Description

result of object detection and shape measurements

`#include <object.h>`

Public Types

- typedef std::list< Pixel > **Pixlist**
- typedef Pixlist::const_iterator **Pixiter**

Public Member Functions

- bool **operator==** (const Object &other)
- bool **operator!=** (const Object &other)
- void **addpixel** (const int x, const int y, const float v)
- int **pixnum** () const
- bool **evaluate** (const float mag_zero, const float saturation)
- bool **evalgood** () const
- void **calc_lum** ()
- void **calc_aper** ()
- void **calc_ir1** ()
- void **drawpixels** (MImage &image)
- void **markpixels** (MMask &image)
- void **setpixels** (MImage &image, const float v)
- float **pixelmedian** ()

Public Attributes

- Pixlist **pixels_**
- bool **evalgood_**
- int **pixnum_**
- int **mx_**
- int my_

  coordinates of maximum intensity

- float mv_

  max intensity

- int **x1_**
- int **y1_**
- int **x2_**
- int y2_

  minimal and maximal extents

- float **icx_**
- float icy_

first moments

- float **ixx_**
- float **iyy_**
- float **ixy_**
- float asym_

  second moments

- float **ir1_**
- float fwhm_

  intens. weighted radius

- float **a_**
- float **b_**
- float pa_

  semi-major and minor axes, pos-angel

- float **lum_**
- float isomag_

  luminosity and isophotal magnitude

- float **isolum_**
- float corelum_

  lum at detection isophote, core lum

- bool saturated_

  at least one pixel saturated

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

- het_lib/photometry/include/object.h
- het_lib/photometry/src/object.cpp

9.535 wfs_photometry::WFSCentroid::object_t Struct Reference

Public Attributes

- int **objectNumber**
- double **X**
- double **cenX**
- double **DX**
- double **Y**
- double **cenY**
- double **DY**
- double **weight**

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

- het_lib/wfs_lib/include/wfs_photometry.h

9.536 ObscGeom Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual void **newData** (int size)
- virtual void **deleteData** (void)
- void **operator=** (ObscGeom &rhs)
- void **Merge** (const ObscGeom &obj1, const ObscGeom &obj2)

Public Attributes

- ushort **Ngeom**
- ushort * **Nvertx**
- float ** **Xgeom_vertx**
- float ** **Ygeom_vertx**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.537 OCDStatus Class Reference

Classes

- struct status

Public Types

- enum **RunShotState_t** {

  RUN_SHOT_UNKNOWN, **RUN_SHOT_IDLE**, **RUN_SHOT_STARTED**, **RUN_SHOT_-SETUP**,

  RUN_SHOT_SETUP_DONE, **RUN_SHOT_EXPOSING** }

Public Member Functions

- bool **initialize** (Payload &payload, void ∗pPrevStatus=NULL)

Public Attributes

- struct OCDStatus::status **m_status**

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

- gui/include/ocdStatusData.h
- gui/src/ocdStatusData.cpp

9.538 offset_fiducial Class Reference

9.538.1 Detailed Description

Offsets the metrology loop's fiducial either by telecentric angle or by change in equatorial coordinates. Parameters are expected to be in arcseconds.

Required parameters:

compensate<bool> - Flag indicating the other probes that are setup should be moved sympathetic to the centering motion of this probe. Defaults to true.

Optional parameters:

dx_asec<double> - Change of fiducial in arcseconds of x.

dy_asec<double> - Change of fiducial in arcseconds of y.

dra_asec<double> - Change of fiducial in arcseconds of RA.

ddec_asec<double> - Change of fiducial in arcseconds of Dec.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **offset_fiducial** (GuideLoop *l, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.538.2 Member Function Documentation

9.538.2.1 void offset_fiducial::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.539 offset_focus Class Reference

9.539.1 Detailed Description

Add the given offset to the the distance held along the normal between the primary and focal spheres.

Optional parameters:

dw<double> - Focus offset to be applied.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **offset_focus** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.539.2 Member Function Documentation

9.539.2.1 void offset_focus::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.540 offset_itf Class Reference

9.540.1 Detailed Description

Applies the given delta(s) to the ITF zero point assumed for all subsequent points sent to the tracker. When called multiple times, the deltas are additive. When called while on trajectory, the change in the zero point is applied immediately. When called while off trajectory, the tracker is moved from the current position to that position plus the given deltas. In either case, the change to the ITF zero point persists until reset or the control system is restarted. At least one delta is required.

See also:

reset_itf_offset, offset_sky, offset_focus.

Optional parameters:

dx<double> - Change in ITF x.

dy<double> - Change in ITF y.

dz<double> - Change in ITF z.

drho<double> - Change in ITF rho.

dtheta<double> - Change in ITF theta.

dphi<double> - Change in ITF phi.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **offset_itf** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.540.2 Member Function Documentation

9.540.2.1 void offset_itf::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.541 offset_probe Class Reference

9.541.1 Detailed Description

Offset a probe.

Required parameters:

dx_ang<double> - On sky offset in x telecentric angle in arcsec from current probe position.

dy_ang<double> - On sky offset in y telecentric angle in arcsec from current probe position.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **offset_probe** (TCSDriver *d, PFIPClient &pfip, Instrument &instrument, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.541.2 Member Function Documentation

9.541.2.1 void offset_probe::process (Message * *message*) [virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/pfip_client.h
- tcs/pfip_client.cpp

9.542 offset_sky Class Reference

9.542.1 Detailed Description

Adds the given angular offset on sky to the current offset applied to each position sent to the tracker. The change to the angular offset persists until reset. At least one delta is required.

See also:

reset_sky_offset, offset_itf, offset_focus.

Optional parameters:

dx_ang<double> - Angular offset, in degrees, to apply along the tracker X axis.

dy_ang<double> - Angular offset, in degrees, to apply along the tracker Y axis.

move_probes<bool> - Flag indicating that the movable probes should be adjusted to account for the requested offset.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **offset_sky** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.542.2 Member Function Documentation

9.542.2.1 void offset_sky::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.543 offset_tiptilt Class Reference

9.543.1 Detailed Description

Adds the given tip and tilt offsets to the current trajectory.

Optional parameters:

dtheta<double> - Offset in degrees of theta.

dphi<double> - Offset in degrees of phi.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **offset_tiptilt** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.543.2 Member Function Documentation

9.543.2.1 void offset_tiptilt::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.544 offset_trajectory Class Reference

9.544.1 Detailed Description

Recalculates the current trajectory based on the given offset. Offsets may be given in either delta RA,Dec or delta X,Y, angular units of arc seconds in either case. When offseting by delta RA,Dec, the RA,Dec of the current trajectory (not current pointing) is adjusted. When offsetting by delta X,Y, the trajectory is recalculated to the RA,Dec of the current pointing plus the given offset. Ideally, prior to calling this method, the trajectory will have already been recalculated based on the current structure azimuth and elevation, in which case these two calling conventions have the same semantics. If the offset is flagged as a correction, then the old trajectory is simply shifted by a delta that remains static over time. The reported pointing coordinates are adjusted by the given correction to reflect the original pointing prior to correction.

Optional parameters:

dra<double> - The desired offset in arc seconds of right ascension. (arcsecond)

ddec<double> - The desired offset in arc seconds of declination. (arcsecond)

dx_ang<double> - The desired offset in arc seconds aligned to the X axis of the focal surface. (arcsecond)

dy_ang<double> - The desired offset in arc seconds aligned to the Y axis of the focal surface. (arcsecond)

adjust_probes<bool> - Flag indicating that the movable probes should be adjusted to account for the requested offset.

correction<bool> - Flag indicating that this offset is a correction to the current trajectory, rather than an offset to a new trajectory.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **offset_trajectory** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.544.2 Member Function Documentation

9.544.2.1 void offset_trajectory::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.545 OModalMatrix Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual void **newData** (int size1, int size2)
- virtual void **deleteData** (void)

Public Attributes

- ushort **width**
- ushort **height**
- ushort **maxterm**
- gsl_matrix * **Omodal**
- gsl_matrix * **O2Zmat**
- gsl_matrix * **Z2Omat**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.546 OnLine Class Reference

Check to see if the legacy dome sub system is connected.

9.546.1 Detailed Description

Check to see if the Structure legacy server is connected.

Check to see if the legacy shutter sub system is connected.

Parameters:

none Check to see if the specified pipeline (camera) is available. The pipeline (camera) name, is prepended to the handler name (i.e. Guider1_).

```
#include <legacyDomeSubSystem.h>
```

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

- legacy/include/legacyDomeSubSystem.h

9.547 Open Class Reference

Open the shutter.

9.547.1 Detailed Description

Parameters:

none

`#include <legacyShutterSubSystem.h>`

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

- legacy/include/legacyShutterSubSystem.h

9.548 camra_handlers::open_shutter Class Reference

Public Member Functions

- **open_shutter** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.548.1 Member Function Documentation

9.548.1.1 void camra_handlers::open_shutter::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.549 open_shutter Class Reference

9.549.1 Detailed Description

Open the configured shutter. This may be a TTL driven lab shutter or the PFIP shutter so exercise caution.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.550 OpenShutter Class Reference

9.550.1 Detailed Description

Open the focal plane shutter.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **OpenShutter** (Receiver ∗rcv, ACQ_SubSystem ∗acq)
- virtual void HandlerAction (Message ∗pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message ∗pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ACQShutterHandlers.cpp

9.551 camra_handlers::override_ifu_associations Class Reference

Public Member Functions

- **override_ifu_associations** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.551.1 Member Function Documentation

9.551.1.1 void camra_handlers::override_ifu_associations::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.552 override_ifu_associations Class Reference

9.552.1 Detailed Description

Sets one or more of the following parameters for a given controller ID: IFU ID, IFU slot, and/or spectrograph slot. Note that there is no check to ensure that the values given are unique with respect to the rest of the array configuration. Note that this is a transient change and the configuration will return to the values read from the command line config upon restart.

Required parameters:

cntl<int> - Controller ID to operate on.

Optional parameters:

ifu_id<string> - New IFU ID for the given controller.

ifu_slot<int> - New IFU slot for the given controller.

spectrograph_slot<string> - New spectrograph slot for the given controller.

```
#include <camra_handler_doc.h>
```

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

- camra/camra_handler_doc.h

9.553 P Class Reference

Public Member Functions

- void process (Message *m)

9.553.1 Member Function Documentation

9.553.1.1 void P::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- playground/toy_subsystem/toy_subsystem.cpp

9.554 ParmsIterator Class Reference

9.554.1 Detailed Description

Iterator class used to iterate over payload parameters. An instance of the iterator is invalidated, resulting in undefined behavior, when set_parm() is called on the corresponding Payload.

`#include <payload.h>`

Public Member Functions

- **ParmsIterator** (const ParmsIterator &i)
- ParmsIterator & **operator++** ()
- bool **operator==** (const ParmsIterator &i)
- bool **operator!=** (const ParmsIterator &i)
- bool **get_parm** (std::string &key, std::string &value, std::string &type)

Friends

- class Payload

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

- het_lib/messaging/include/payload.h
- het_lib/messaging/src/payload.cpp

9.555 ParmsObject Class Reference

9.555.1 Detailed Description

Interface class to serializing object to Payloads. Implement apply_parms() to set the child classes parameters in the given Payload. This allows objects to be serialized to both Event and Message wire data structures.

`#include <payload.h>`

Public Member Functions

- virtual void **apply_parms** (Payload &p) const =0

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

- het_lib/messaging/include/payload.h

9.556 PASCameraClient Class Reference

9.556.1 Detailed Description

A class that encapsulates the Camera Subsystem interface on the PAS. Used for the guide probes, wavefront sensors, the tip-tilt camera and the distance-measuring interferometer.

`#include <pas_camera.h>`

Public Member Functions

- **PASCameraClient** (const std::string &endpoint, const std::string &camName)
- bool **CameraOnline** ()
- bool **StartPipeline** (bool do_exception=0)
- bool **StopPipeline** ()
- void **SetProcessImages** (bool state)
- void **SetAnalysisRegion** (int xmin, int xmax, int ymin, int ymax)
- AnalysisRegion **GetAnalysisRegion** (bool do_exception=0)

Protected Attributes

- Sender * s_

  the sender connected to this camera

- const std::string name_

  the name of the PAS Pipeline we represent

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

- tcs/pas_camera.h
- tcs/pas_camera.cpp

9.557 pasConfig Class Reference

Public Member Functions

- **pasConfig** (bool isNamed=0)
- void **parsePipelineConfigs** (void)
- void set_options (OptionParser &optionParser)

  Adds options to default set.

- void print_summary (void)

  Called prior to displaying options.

Public Attributes

- int m_numPipelines

  Number of pipeline(s).

- string m_pipeline [24+1]

  --pipelines option - list of pipeline(s)

- PipelineConfig * m_pPipelineConfig [24]

  Pipeline configuration(s).

9.557.1 Member Function Documentation

9.557.1.1 void pasConfig::print_summary (void) [virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.557.1.2 void pasConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- pas/include/pasConfig.h
- pas/src/pasConfig.cpp

9.558 pasDevicePipelineSubSystem Class Reference

Classes

- struct DMIData
- struct SAMSData

Public Member Functions

- **pasDevicePipelineSubSystem** (PipelineConfig *pConfig)
- virtual void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- virtual void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- virtual bool **SetGROCHandler** (Message *pMessage, Message **ppReply, double groc)
- virtual bool **StatusHandler** (Message *pMessage, Message **ppReply, bool update_state)
- virtual bool **EnableSACDMIHandler** (Message *pMessage, Message **ppReply, bool state)
- virtual bool **EnableCCASDMIHandler** (Message *pMessage, Message **ppReply, bool state)

Public Attributes

- class DevicePipelineSetGROCHandlerClass * **m_SetGROCHandler**
- class DevicePipelineStatusHandlerClass * **m_StatusHandler**
- class DevicePipelineEnableSACDMIHandlerClass * **m_EnableSACDMIHandler**
- class DevicePipelineEnableCCASDMIHandlerClass * **m_EnableCCASDMIHandler**

Protected Types

- enum **Cmd_t** {

  GET_CCAS_DISTANCE, **GET_SAC_DISTANCE**, **GET_GROC**, **SET_GROC**,

  GET_SAMS_DATA }

Protected Member Functions

- virtual bool **initialize** (void)
- virtual bool **runPipeline** (ProcessingMode_t procMode, Message **ppReply=NULL, const char *saveImageFilename=NULL)
- void **constructDMIPipeline** (void)
- void **constructSAMSPipeline** (void)
- void **setupDMICmds** (void)
- void **setupSAMSCmds** (void)
- virtual double **statusWorker** (bool bForceEvent=0)
- virtual bool **checkReply** (int cmd, unsigned int replyLength, unsigned char *replyString)
- virtual bool **getStatus** (bool retry=0, bool doException=0, bool extraBool=0)
- virtual void **populateStatusPayload** (Payload &payload)

Protected Attributes

- std::map< Cmd_t, std::string > **m_Cmd**
- DMIData **m_DMIData**
- SAMSData **m_SAMSData**
- DMIData **m_lastDMIData**
- SAMSData **m_lastSAMSData**
- int **m_lastPipelineCycleTime_ms**
- bool **m_bLastPipelineRunning**

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

- pas/include/pasDevicePipelineSubSystem.h
- pas/src/pasDevicePipelineSubSystem.cpp

9.559 PASException Class Reference

Public Member Functions

- **PASException** (tcsexcept::tcs_exception_t __pas, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.560 pasImageCleanerStandard Class Reference

Public Member Functions

- **pasImageCleanerStandard** (PipelineConfig *pConfig)
- virtual void **initialize** (void)
- virtual bool **clean** (const int filterIndex, const CleaningBiasMode_t biasMode, ExposureInfo *pExposureInfo)

Protected Member Functions

- bool **loadFlats** (void)
- bool **loadHotPixels** (void)
- bool **loadMaskDefinition** (void)
- bool **buildBiasAreaList** (void)
- void **maskBiasAdjust** (const CleaningBiasMode_t biasMode, ExposureInfo *pExposureInfo)
- void **overScanBiasAdjust** (const CleaningBiasMode_t biasMode, ExposureInfo *pExposureInfo)
- void **removeHotPixels** (ExposureInfo *pExposureInfo)
- void **flatten** (const int filterIndex, ExposureInfo *pExposureInfo)

Protected Attributes

- int **m_nAxes** [2]
- unsigned int **m_numMaskVertices**
- Point_t * **m_pMaskVertex**
- double **m_biasAdjustValue**
- unsigned int **m_numBiasPixels**
- Pixel_t * **m_pBiasAreaList**
- unsigned int **m_numHotPixels**
- HotPixel_t * **m_pHotPixel**
- MArray< float, 2 > * **m_pFlatImage** [5]

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

- pas/include/pasImageCleanerStandard.h
- pas/src/pasImageCleanerStandard.cpp

9.561 pasImagePipelineSubSystem Class Reference

Classes

- struct FITSMapEntry
- struct ImageHandlerData
- struct PFIPData
- struct TCSData

Public Types

- enum **FITSItemType_t** { **STRING**, **INTEGER**, **DOUBLE**, **BOOLEAN** }
- typedef std::map< string, FITSMapEntry > **EventParmToFITSMappingTable_t**

Public Member Functions

- **pasImagePipelineSubSystem** (PipelineConfig *pConfig)
- virtual void **start** (bool takeControl=0, bool checkSocketConnection=0)
- virtual void **stop** (bool relinquishControl=0)
- virtual void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- virtual void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- TCSClient::TelescopeData * **getTelescopeData** ()
- void **ReceiveEvents** (void)
- Camera * **getCamera** (void)
- tcs_photometry::GuideRect **getGuideRect** (void)
- int **getCameraFilterIndex** (void)
- **HANDLER_DECL** (ImagePipeline, StartSendingImages)
- **HANDLER_DECL** (ImagePipeline, StopSendingImages)
- virtual bool **SetPixelFormatHandler** (Message *pMessage, Message **ppReply, string type)
- virtual bool **SetReadoutSpeedHandler** (Message *pMessage, Message **ppReply, string speed)
- virtual bool **SetImageTypeHandler** (Message *pMessage, Message **ppReply, string type)
- virtual bool **SetAnalysisRegionHandler** (Message *pMessage, Message **ppReply, int xmin, int xmax, int ymin, int ymax)
- **HANDLER_DECL** (ImagePipeline, GetAnalysisRegion)
- virtual bool **SetRegionOfInterestHandler** (Message *pMessage, Message **ppReply, int xmin, int xmax, int ymin, int ymax)
- **HANDLER_DECL** (ImagePipeline, GetRegionOfInterest)
- virtual bool **SetExposureTimeHandler** (Message *pMessage, Message **ppReply, double time)
- **HANDLER_DECL** (ImagePipeline, GetExposureTime)
- virtual bool **SetOverScanHandler** (Message *pMessage, Message **ppReply, int overScan)
- **HANDLER_DECL** (ImagePipeline, GetOverScan)
- virtual bool **SetFilterExposureTimeFactorHandler** (Message *pMessage, Message **ppReply, string filterName, double factor)
- virtual bool **SetFilterZeroPointHandler** (Message *pMessage, Message **ppReply, string filterName, double zeroPoint)
- virtual bool **SetExposuresPerCycleHandler** (Message *pMessage, Message **ppReply, int exposures)
- virtual bool **SetFilterSequenceHandler** (Message *pMessage, Message **ppReply, string sequence)
- virtual bool **SetBinningHandler** (Message *pMessage, Message **ppReply, int width, int height)

- **HANDLER_DECL** (ImagePipeline, GetGain)
- virtual bool **SetTargetTemperatureHandler** (Message *pMessage, Message **ppReply, double temp)
- **HANDLER_DECL** (ImagePipeline, GetTemperatures)
- virtual bool **SetNumFlushesHandler** (Message *pMessage, Message **ppReply, int flushes)
- virtual bool **SetObjectAndMagnitudesHandler** (Message *pMessage, Message **ppReply, string object)
- virtual bool **SetLightSourcePowerHandler** (Message *pMessage, Message **ppReply, bool state)
- virtual bool **SetLightSourcePowerLevelHandler** (Message *pMessage, Message **ppReply, double level)
- **HANDLER_DECL** (ImagePipeline, GetLightSourcePowerLevel)
- virtual bool **SetLightSourcePowerHoldTimeHandler** (Message *pMessage, Message **ppReply, int time)
- virtual bool **SetCleanImagesHandler** (Message *pMessage, Message **ppReply, bool clean)
- virtual bool **SetProcessImagesHandler** (Message *pMessage, Message **ppReply, bool process)
- virtual bool **SetStoreImagesHandler** (Message *pMessage, Message **ppReply, bool store)
- virtual bool **SetCleaningBiasTypeHandler** (Message *pMessage, Message **ppReply, string type)

- virtual bool **SetProcessingTypeHandler** (Message *pMessage, Message **ppReply, string type)
- **HANDLER_DECL** (ImagePipeline, GetPhotPhotometryOptions)
- **HANDLER_DECL** (ImagePipeline, GetWFSPhotometryOptions)
- virtual bool **SetPhotPhotometryOptionsHandler** (Message *pMessage, Message **ppReply, double mag_zero, double detect_sigma, int min_pixels, int max_pixels, double nan_value, double saturation, double gain, int sky_order, int sky_annulus, double kron_aperture, double fixed_aperture)

- virtual bool **SetWFSPhotometryOptionsHandler** (Message *pMessage, Message **ppReply, bool bRenewFlag, int pixelsAcrossSubAperture, int pixelsPerSubAperture, int numZernikeModes, int numZernikeModesToReconstruct, double diskX, double diskY, double diskRadius, double pixelScale, double pupilRadius, double rotation, double refWaveLength)
- void **getTelescopeDataFromFitsHeader** (FitsIn *pFitsIn)
- void **printTelescopeData** (void)

Public Attributes

- class ImagePipelineSetPixelFormatHandlerClass * **m_SetPixelFormatHandler**
- class ImagePipelineSetReadoutSpeedHandlerClass * **m_SetReadoutSpeedHandler**
- class ImagePipelineSetImageTypeHandlerClass * **m_SetImageTypeHandler**
- class ImagePipelineSetAnalysisRegionHandlerClass * **m_SetAnalysisRegionHandler**
- class ImagePipelineSetRegionOfInterestHandlerClass * **m_SetRegionOfInterestHandler**
- class ImagePipelineSetExposureTimeHandlerClass * **m_SetExposureTimeHandler**
- class ImagePipelineSetOverScanHandlerClass * **m_SetOverScanHandler**
- class ImagePipelineSetFilterExposureTimeFactorHandlerClass * **m_SetFilterExposureTimeFactorHandler**
- class ImagePipelineSetFilterZeroPointHandlerClass * **m_SetFilterZeroPointHandler**
- class ImagePipelineSetExposuresPerCycleHandlerClass * **m_SetExposuresPerCycleHandler**
- class ImagePipelineSetFilterSequenceHandlerClass * **m_SetFilterSequenceHandler**
- class ImagePipelineSetBinningHandlerClass * **m_SetBinningHandler**
- class ImagePipelineSetTargetTemperatureHandlerClass * **m_SetTargetTemperatureHandler**
- class ImagePipelineSetNumFlushesHandlerClass * **m_SetNumFlushesHandler**
- class ImagePipelineSetObjectAndMagnitudesHandlerClass * **m_SetObjectAndMagnitudesHandler**

- class ImagePipelineSetLightSourcePowerHandlerClass * **m_SetLightSourcePowerHandler**
- class ImagePipelineSetLightSourcePowerLevelHandlerClass * **m_SetLightSourcePowerLevelHandler**
- class ImagePipelineSetLightSourcePowerHoldTimeHandlerClass * **m_SetLightSourcePowerHoldTimeHandler**
- class ImagePipelineSetCleanImagesHandlerClass * **m_SetCleanImagesHandler**
- class ImagePipelineSetProcessImagesHandlerClass * **m_SetProcessImagesHandler**
- class ImagePipelineSetStoreImagesHandlerClass * **m_SetStoreImagesHandler**
- class ImagePipelineSetCleaningBiasTypeHandlerClass * **m_SetCleaningBiasTypeHandler**
- class ImagePipelineSetProcessingTypeHandlerClass * **m_SetProcessingTypeHandler**
- class ImagePipelineSetPhotometryOptionsHandlerClass * **m_SetPhotometryOptionsHandler**
- class ImagePipelineSetPhotPhotometryOptionsHandlerClass * **m_SetPhotPhotometryOptionsHandler**
- class ImagePipelineSetWFSPhotometryOptionsHandlerClass * **m_SetWFSPhotometryOptionsHandler**
- tthread::mutex **accessMutex**
- bool **bCleaningConfigured**
- bool **bCleaningEnabled**
- CleaningMode_t **cleaningMode**
- CleaningBiasMode_t **cleaningBiasMode**
- bool **bProcessingConfigured**
- bool **bProcessingEnabled**
- ProcessingMode_t **processingMode**
- bool **bStoringConfigured**
- bool **bStoringEnabled**
- int **lastPipelineCycleTime_ms**
- int **lastImagesPerCycle**
- bool **bLastCleaningConfigured**
- bool **bLastCleaningEnabled**
- bool **bLastProcessingConfigured**
- bool **bLastProcessingEnabled**
- ProcessingMode_t **bLastProcessingMode**
- bool **bLastStoringConfigured**
- bool **bLastStoringEnabled**

Protected Member Functions

- void **ReceiveEventsThreadEntryPoint** (void)
- virtual bool **runPipeline** (ProcessingMode_t procMode, Message **ppReply=NULL, const char *saveImageFilename=NULL)
- virtual double **statusWorker** (bool bForceEvent=0)
- bool **turnOnLightSource** (void)
- bool **takeAnExposure** (void)
- bool **cleanImage** (CleaningMode_t cleanMode, CleaningBiasMode_t biasMode, Message **ppReply)
- bool **processImage** (ProcessingMode_t procMode, Message **ppReply=NULL)
- bool **storeImage** (bool bCoreMatch, const char *imageFilename)
- bool **sendImage** (void)
- void **constructCameraPipeline** (void)
- bool **runCameraPipeline** (void)

- bool **startupPipeline** (void)
- bool **shutdownPipeline** (void)
- void addTCSDataToFITSFile (TCSClient::TelescopeData &data, std::string cameraTypeName, pas::Camera_t cameraType, double plateScaleX, double plateScaleY, unsigned int wcsRefPixelX, unsigned int wcsRefPixelY, FitsOutWrapper *pFitsOut)
- void **addPFIPEventDataToFITSFile** (PFIPData &data, pas::Camera_t cameraType, FitsOutWrapper *pFitsOut)
- void **addTCSEventDataToFITSFile** (TCSData &data, pas::Camera_t cameraType, FitsOutWrapper *pFitsOut)
- void **addMetrologyConfigurationToFITSFile** (void)
- void **addMetrologyDataToFITSFile** (Payload &eventPayload)
- void **decipher_and_write_lamp** (FitsOutWrapper *pFitsOut, const string &lamp)
- void **mapPayloadToFITSFile** (Payload &payload, EventParmToFITSMappingTable_t &mapingTable)
- void **setupMetrologyDataToFITSKeywordMap** (void)
- void **setupMetrologyConfigurationToFITSKeywordMap** (void)

Protected Attributes

- LightSource * **m_pLightSource**
- Camera * **m_pCamera**
- bool **m_bSimulatedCamera**
- ImageProcessor * **m_pProcessor**
- ImageCleaner * **m_pCleaner**
- pasImageStorage * **m_pStorage**
- ImageHandlerData **m_ImageHandlerData**
- TCSClient::TelescopeData **m_telescopeData**
- struct pasImagePipelineSubSystem::PFIPData **m_PFIPData**
- struct pasImagePipelineSubSystem::TCSData **m_TCSData**
- struct {
  tthread::mutex **accessMutex**
  bool **bCleaningConfigured**
  bool **bCleaningEnabled**
  CleaningMode_t **cleaningMode**
  CleaningBiasMode_t **cleaningBiasMode**
  bool **bProcessingConfigured**
  bool **bProcessingEnabled**
  ProcessingMode_t **processingMode**
  bool **bStoringConfigured**
  bool **bStoringEnabled**
  int **lastPipelineCycleTime_ms**
  int **lastImagesPerCycle**
  bool **bLastCleaningConfigured**
  bool **bLastCleaningEnabled**
  bool **bLastProcessingConfigured**
  bool **bLastProcessingEnabled**
  ProcessingMode_t **bLastProcessingMode**
  bool **bLastStoringConfigured**
  bool **bLastStoringEnabled**
  } **m_pipelineStatus**

- tthread::mutex **m_GuideRectMutex**
- tcs_photometry::GuideRect **m_GuideRect**
- string **m_ObjectId**
- string **m_Magnitudes**
- EventSink * **m_pReceiveEventsSink**
- tthread::thread * **m_pReceiveEventsThread**
- volatile bool **m_bReceiveEventsThreadRunning**
- volatile bool **m_bStopReceiveEventsThread**
- EventParmToFITSMappingTable_t **m_metrologyDataToFITSKeyword**
- EventParmToFITSMappingTable_t **m_metrologyConfigurationToFITSKeyword**

Friends

- class pasImageProcessorWFS

9.561.1 Member Function Documentation

9.561.1.1 void pasImagePipelineSubSystem::addTCSDataToFITSFile (TCSClient::TelescopeData & *data*, std::string *cameraTypeName*, pas::Camera_t *cameraType*, double *plateScaleX*, double *plateScaleY*, unsigned int *wcsRefPixelX*, unsigned int *wcsRefPixelY*, FitsOutWrapper * *pFitsOut*) [protected]

Add a linear wcs to a 2D image.

ra_deg and dec_deg are the coordinates of the pixel {cx,cy} in decimal degrees. parallacticAngle is the position angle on the sky in astronomical convention (reverse from the usual mathematical convention on angles) pixscale_x and pixscale_y are the extent of a pixel in degrees in the CCD x and y directions

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

- pas/include/pasImagePipelineSubSystem.h
- pas/src/pasImagePipelineSubSystem.cpp

9.562 pasImageProcessorPhot Class Reference

Public Member Functions

- **pasImageProcessorPhot** (PipelineConfig *pConfig)
- virtual bool **process** (ProcessingMode_t processingMode, ExposureInfo *pExposureInfo, pasImagePipelineSubSystem *pSubSystem, Payload &metrologyEventPayload)
- virtual void **getPhotOptions** (tcs_photometry::ConfigureParams &configParams)
- virtual void **getPhotOptionsToPayload** (Payload &payload)
- virtual void **setPhotOptions** (const tcs_photometry::ConfigureParams &configParams)
- virtual float getCurrentPupilIllumination (const TCSClient::TelescopeData *pTelescopeData)

  calculate the illuminated fraction of the pupil given current tracker and focal plane coordinates

Protected Attributes

- tcs_photometry::Photometry * **m_pPhotometry**
- HETillumination **m_hi**

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

- pas/include/pasImageProcessorPhot.h
- pas/src/pasImageProcessorPhot.cpp

9.563 pasImageProcessorWFS Class Reference

Public Member Functions

- **pasImageProcessorWFS** (PipelineConfig *pConfig)
- virtual bool **process** (ProcessingMode_t processingMode, ExposureInfo *pExposureInfo, pasImagePipelineSubSystem *pSubSystem, Payload &metrologyEventPayload)
- virtual void **getWFSOptions** (StaticConfig &configParams)
- virtual void **getPhotOptionsToPayload** (Payload &payload)
- virtual void **setWFSOptions** (const StaticConfig &configParams)
- virtual void **getGraphicOverlayData** (unsigned char **wfs_shgrid, size_t &wfs_shgridSize, unsigned char **wfs_m1array, size_t &wfs_m1arraySize, unsigned char **wfs_pupil, size_t &wfs_pupilSize, unsigned char **wfs_hetobsc, size_t &wfs_hetobscSize, unsigned char **wfs_shtrkdat, size_t &wfs_shtrkdatSize)

Protected Attributes

- wfslib_het * **m_pWFSLib**

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

- pas/include/pasImageProcessorWFS.h
- pas/src/pasImageProcessorWFS.cpp

9.564 pasImageStorage Class Reference

Base Image Storage class.

`#include <pasImageStorage.h>`

Public Member Functions

- **pasImageStorage** (PipelineConfig *pConfig)
- virtual bool **store** (uint8_t *pFITSFileBuffer, unsigned int FITSBufferSize, const char *cameraTypeName, Camera_t cameraType, Image_t imageType, bool bInCaseOfCrash, const char *imageFilename)=0
- void **createStorageDirectory** (Camera_t cameraType, Image_t imageType)

Protected Attributes

- PipelineConfig * **m_pConfig**
- std::string **m_coreMatchFileName**
- std::string **m_storageDir**

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

- pas/include/pasImageStorage.h
- pas/src/pasImageStorage.cpp

9.565 pasImageStorageFITS Class Reference

Public Member Functions

- **pasImageStorageFITS** (PipelineConfig *pConfig)
- virtual bool **store** (uint8_t *pFITSFileBuffer, unsigned int FITSFileBufferSize, const char *cameraTypeName, Camera_t cameraType, Image_t imageType, bool bCoreMatch, const char *imageFilename=NULL)

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

- pas/include/pasImageStorageFITS.h
- pas/src/pasImageStorageFITS.cpp

9.566 pasPipelineSubSystem Class Reference

Public Member Functions

- **pasPipelineSubSystem** (PipelineConfig *pConfig)
- virtual void **start** (bool takeControl=0, bool checkSocketConnection=0)
- virtual void **stop** (bool relinquishControl=0)
- virtual void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- virtual void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- PipelineConfig * **getConfig** (void)
- void **startPipelineThread** (ProcessingMode_t procMode, Message **ppReply=NULL, const char *saveImageFilename=NULL)
- void **checkPreviousStopComplete** (void)
- **HANDLER_DECL** (Pipeline, OnLine)
- virtual bool **SetCycleTimeHandler** (Message *pMessage, Message **ppReply, double seconds)
- virtual bool **SetIntervalSpeedHandler** (Message *pMessage, Message **ppReply, double speed)
- **HANDLER_DECL** (Pipeline, StartPipeline)
- **HANDLER_DECL** (Pipeline, StopPipeline)
- **HANDLER_DECL** (Pipeline, TakeControl)
- **HANDLER_DECL** (Pipeline, RelinquishControl)
- virtual bool **CyclePipelineHandler** (Message *pMessage, Message **ppReply, string procMode, string imageFilename)
- **HANDLER_DECL** (Pipeline, ForceStatusEvent)

Public Attributes

- class PipelineSetCycleTimeHandlerClass * **m_SetCycleTimeHandler**
- class PipelineSetIntervalSpeedHandlerClass * **m_SetIntervalSpeedHandler**
- class PipelineCyclePipelineHandlerClass * **m_CyclePipelineHandler**

Protected Member Functions

- void **pipelineThreadEntryPoint** (void)
- void **emitPipelineTimingEvent** (void)
- virtual bool **runPipeline** (ProcessingMode_t procMode, Message **ppReply, const char *saveImageFilename)=0
- virtual bool **startupPipeline** (void)
- virtual bool **shutdownPipeline** (void)

Protected Attributes

- Pipeline_t **m_pipelineType**
- bool **m_bSimulatedCamera**
- CleaningMode_t **m_cleaningMode**
- CleaningBiasMode_t **m_cleaningBiasMode**
- ProcessingMode_t **m_processingMode**
- PipelineConfig * **m_pConfig**
- tthread::mutex **m_pipelineThreadMutex**
- tthread::thread * **m_pPipelineThread**

- tthread::Timer **m_pipelineThreadCycleTimer**
- Message ** **m_pipelineThreadppReply**
- const char * **m_pipelineThreadSaveImageFilename**
- ProcessingMode_t **m_pipelineThreadProcessingMode**
- volatile bool **m_bPipelineThreadRunning**
- volatile bool **m_bPipelineThreadStopRequested**
- volatile bool **m_bPipelineThreadStopped**
- volatile bool **m_bLastPipelineThreadRunning**

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

- pas/include/pasPipelineSubSystem.h
- pas/src/pasPipelineSubSystem.cpp

9.567 pasServer Class Reference

Public Member Functions

- **pasServer** (const string &appName, pasConfig *pConfig)
- virtual int **start** (void)
- virtual void **stop** (void)
- pasConfig * **GetConfig** ()

Public Attributes

- TCSClient * **m_pTCSClient**
- WeatherClient * **m_pWeatherClient**

Protected Member Functions

- void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- void **SetName** (string name)
- const char * **GetName** (void)
- void **SetDebug** (bool on)

Protected Attributes

- string **m_Name**
- bool **m_Debug**
- pasConfig * **m_pConfig**
- Receiver * **m_pReceiver**
- Sender * **m_pLegacySender**
- pasPipelineSubSystem * **m_PipelineSubSystem** [24]

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

- pas/include/pasServer.h
- pas/src/pasServer.cpp

9.568 liborchestrate::PasSubsystem Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.569 pasTTLaserLightSource Class Reference

Public Member Functions

- **pasTTLaserLightSource** (LightSourceConfig *pConfig)
- virtual void **setPowerMonitorCycleTime_ms** (int cycleTime_ms)
- virtual int **getPowerMonitorCycleTime_ms** (void)
- virtual bool **powerOn** (bool bChangePowerLevel=0)
- virtual bool **powerOff** (void)
- virtual bool **isPowerOn** (void)
- virtual bool **setPowerHoldTime** (int holdTime)
- virtual bool **setPowerLevel** (double powerLevel)
- virtual bool **getPowerLevel** (double *pPowerLevel)
- virtual bool **atFullPower** (void)

Protected Member Functions

- bool **initialize** (void)
- bool **sendCmdCheckReply** (char *cmdStr, unsigned int timeout_sec=5)
- virtual void **PowerMonitor** (void)

Protected Attributes

- std::string **m_Route**
- bool **m_EmptySocketBeforeNextCmd**
- char **m_Response** [256+1]
- int **m_PowerMonitorCycleTime_ms**
- tthread::ThreadWorker * **m_PowerMonitorThread**
- tthread::mutex **m_lightSourceAccessMutex**
- char **m_ModuleType** [81]
- char **m_SerialNumber** [81]
- char **m_ManuDate** [81]
- char **m_Manufacturer** [81]
- char **m_ManuLocation** [81]
- char **m_ResetTime** [81]
- char **m_ServiceTime** [81]
- bool **m_bPowerOn**
- int **m_nPowerHoldTime_secs**
- double **m_dPowerLevel**
- double **m_dPowerOffAt**

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

- pas/include/pasTTLaserLightSource.h
- pas/src/pasTTLaserLightSource.cpp

9.570 Payload Class Reference

9.570.1 Detailed Description

The Payload class is used to construct messages from Sender to Receiver. Messages (with base class Payload) are in the form of a list of name/value pairs, and the Payload class provides construction and parsing methods.

`#include <payload.h>`

Classes

- class Data

Public Member Functions

- **Payload** (const void *data, size_t size)
- **Payload** (std::string s)
- bool set_parm (const std::string &key, const std::string &value, const std::string &type)
- bool set_data (void *data, size_t size)
- const void * data (void)
- size_t size (void)
- std::string **str** (void)
- ParmsIterator **begin** (void)
- ParmsIterator & **end** (void)
- std::string **format_parms** (const char sep= ' ')
- std::string **format_parms_ignore_meta_data** (const char sep= ' ')

- template<typename T >
  Payload & operator<< (const std::pair< std::string, T > &key_value)
- template<typename T >
  Payload & **operator**<< (const std::pair< char *, T > &key_value)
- template<typename T >
  Payload & **operator**<< (const std::pair< const char *, T > &key_value)
- Payload & **operator**<< (Payload &p)
- Payload & **operator**<< (const ParmsObject &p)

Parameter mutators

Mutators for setting the parameterization of the Payload. The `key value` pair given is stored in the payload and `true` returned.

XXX document error conditions. what if duplicate key?

- virtual bool **set_parm** (const std::string &key, const bool &value)
- virtual bool **set_parm** (const std::string &key, const int &value)
- virtual bool **set_parm** (const std::string &key, const uint &value)
- virtual bool **set_parm** (const std::string &key, const double &value)
- virtual bool **set_parm** (const std::string &key, const std::string &value)
- virtual bool **set_parm** (const std::string &key, const char *value)
- virtual bool **set_parm** (const std::string &key, const fptime &value)

Parameter accessors

Accessors for querying the parameterization of the Payload. The `key` given is mapped to a corresponding value. If found, the `value` is set and `true` returned. Returns `false` otherwise. The string version normalizes values to strings, and an optional data type string reference can be supplied to receive the type, as in ParmsIterator::get_parm().

- bool **get_parm** (const std::string &key, bool &value)
- bool **get_parm** (const std::string &key, int &value)
- bool **get_parm** (const std::string &key, uint &value)
- bool **get_parm** (const std::string &key, double &value)
- bool **get_parm** (const std::string &key, std::string &value)
- bool **get_parm** (const std::string &key, std::string &value, std::string &data_type)
- bool **get_parm** (const std::string &key, fptime &value)

Protected Member Functions

- bool **get_parm_nr** (const std::string &key, double &value)
- bool **set_str** (const std::string &str)

Protected Attributes

- char * **parms_str_**
- cJSON * **parms_json_**
- Data * **data_**
- zmsg_t * **raw_**

Friends

- class Event
- class Message
- class Sender
- class Receiver
- class MetaPayload
- void **receiver_recv_thread** (void *z, zctx_t *ctx, void *pipe)
- int **receiver_shutdown_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **receiver_recv_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)

9.570.2 Member Function Documentation

9.570.2.1 const void * Payload::data (void)

Returns a pointer to the start of the data for the Payload. Returns NULL if the data buffer is uninitialized.

9.570.2.2 template<typename T > Payload& Payload::operator<< (const std::pair< std::string, T > & *key_value*) [inline]

Adds the given templated parameter key/value to the Payload. A reference to the calling Payload is returned.

Reimplemented in Event.

9.570.2.3 bool Payload::set_data (void * *data*, size_t *size*)

Fill the data. This copies the `size` bytes of the memory region pointed to by `data` into the Payload. This deallocates any previous data set for this `Payload`. Returns false if an error occurs or the Payload is immutable.

9.570.2.4 bool Payload::set_parm (const std::string & *key*, const std::string & *value*, const std::string & *type*)

Sets the given key/value pair applying the type specified to the value.

9.570.2.5 size_t Payload::size (void)

Returns the size of the data for the Payload. Returns zero if the data buffer is uninitialized.

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

- het_lib/messaging/include/payload.h
- het_lib/messaging/src/payload.cpp

9.571 pduConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- int **m_numOutlets**
- int **m_numOutletsSpecified**
- string **m_guiTabName**
- int **m_numOutletsInUse**
- string **m_outlet** [16+1]

9.571.1 Member Function Documentation

9.571.1.1 void pduConfig::set_options (OptionParser & *optionParser*) `[virtual]`

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

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

- apc/include/apcConfig.h
- apc/src/apcConfig.cpp

9.572 PduOnLine Class Reference

Check to see if Pdu is running.

9.572.1 Detailed Description

Parameters:

none

`#include <apcPDUSubSystem.h>`

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

- apc/include/apcPDUSubSystem.h

9.573 pfip_clear_positions Class Reference

9.573.1 Detailed Description

Clears any set probe positions so that a subsequent move will not attempt to repositions the probes.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **pfip_clear_positions** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.573.2 Member Function Documentation

9.573.2.1 void pfip_clear_positions::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.574 pfip_close_shutter Class Reference

9.574.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **pfip_close_shutter** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.574.2 Member Function Documentation

9.574.2.1 void pfip_close_shutter::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.575 pfip_coord Class Reference

9.575.1 Detailed Description

For going between telecentric angle to probe angular position - used by PFIP

`#include <coords.h>`

Public Member Functions

- pfip_coord (const pfip::ProbeID ProbeID)
- int **probe2telecen** (const int enc_t, const int enc_p, double &xi_x, double &xi_y)
- int telecen2probe (const double xi_x, const double xi_y, int &enc_t, int &enc_p)
- void **enc2ang** (const int enc_t, const int enc_p, double &beta, double &gma)
- void **ang2enc** (const double beta, const double gma, int &enc_t, int &enc_p)
- void **enc2zpt** (const int enc_t, const int enc_p, double &xpix0, double &ypix0)

Public Attributes

- int **enc_t_min_**
- int **enc_t_max_**
- int **enc_p_min_**
- int **enc_p_max_**
- double **plate_scale**
- double **orientation_offset**
- double **homeOrientation**
- int **naxis1**
- int **naxis2**

Friends

- struct bestDistance

9.575.2 Constructor & Destructor Documentation

9.575.2.1 pfip_coord::pfip_coord (const pfip::ProbeID *ProbeID*)

< radius of probe annulus in degrees on sky

< radius of probe annulus in degrees on sky

9.575.3 Member Function Documentation

9.575.3.1 int pfip_coord::telecen2probe (const double *xi_x*, const double *xi_y*, int & *enc_t*, int & *enc_p*)

< Tolerance to stop the iterations

< Maximum number of iterations

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

- het_lib/coord_trans/include/coords.h
- het_lib/coord_trans/src/pfip.cpp

9.576 PFIP_EPICSVariable Class Reference

Public Attributes

- pfip::EPICSVariableLookupKey **key**
- char **name** [50]
- chtype **dataType**
- unsigned int **dataSize**
- chid **channelId**
- channel_state **status**

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

- pfip/include/EPICSComm.h

9.577 pfip_insert_acquisition_pickoff Class Reference

9.577.1 Detailed Description

Inserts the acquisition camera pickoff mirror.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **pfip_insert_acquisition_pickoff** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.577.2 Member Function Documentation

9.577.2.1 void pfip_insert_acquisition_pickoff::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.578 PFIP_Monitor Class Reference

9.578.1 Detailed Description

A class to manage reading from the PFIP (modbus) PLC

`#include <PFIPMonitor.h>`

Public Member Functions

- PFIP_Monitor (string monitorName, string enclName, int statusFrequency)

  Create the PFIP monitor thread that read alarms/temperatures in PFIP enclosure.

- void checkEnvironment ()

  Handle the notifications received from the PLC device.

9.578.2 Constructor & Destructor Documentation

9.578.2.1 PFIP_Monitor::PFIP_Monitor (string *monitorName*, string *enclName*, int *statusFrequency*) [inline]

Parameters:

configObj[in] - the PFIP Configuration

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

- pfip/include/PFIPMonitor.h
- pfip/src/PFIPMonitor.cpp

9.579 pfip_move_probes Class Reference

9.579.1 Detailed Description

Initiate any staged probe moves. This call blocks until probes are in position.

See also:

set_position

`#include <tcs_handler_doc.h>`

Public Member Functions

- **pfip_move_probes** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.579.2 Member Function Documentation

9.579.2.1 void pfip_move_probes::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.580 pfip_open_shutter Class Reference

9.580.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **pfip_open_shutter** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.580.2 Member Function Documentation

9.580.2.1 void pfip_open_shutter::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.581 pfip_withdraw_acquisition_pickoff Class Reference

9.581.1 Detailed Description

Withdraws the acquisition camera pickoff mirror.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **pfip_withdraw_acquisition_pickoff** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.581.2 Member Function Documentation

9.581.2.1 void pfip_withdraw_acquisition_pickoff::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.582 PFIPClient Class Reference

Public Member Functions

- **PFIPClient** (const std::string &route, const std::string &event_route, const std::string &name, bool shutter_pfip)
- void **open_shutter** (void)
- void **close_shutter** (void)
- void **mark_shutter_close** (void)
- void **write** (FitsOutWrapper &f, bool include_exptime)
- void **get_header_info** (void)
- bool **active** (void)
- **PFIPClient** (const std::string &route, TCSDriver *driver)
- void **open_shutter** (bool do_exception=1)
- void **close_shutter** (bool do_exception=1)
- void **insert_acquisition_pickoff** (bool do_exception=1)
- void **withdraw_acquisition_pickoff** (bool do_exception=1)
- void **update_status** (Event *e)
- void **move** (void)
- void **move_async** (void)
- void **wait_for_last_move** (void)
- void **offset_probes** (const vec2_t &offset_arcsec)
- void **clear_positions** (void)
- Instrument & **guider1** (void)
- Instrument & **guider2** (void)
- Instrument & **wfs1** (void)
- Instrument & **wfs2** (void)
- Instrument & **cwfs** (void)
- Instrument & **acq** (void)
- Instrument & **hpf_acq** (void)
- Instrument & **bib** (void)
- Instrument & **cat** (void)
- Instrument & **vat** (void)
- Instrument & **pv** (void)

Protected Attributes

- tthread::recursive_mutex **position_mutex_**

Friends

- class set_position
- class offset_probe

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

- camra/pfip_client.h
- tcs/pfip_client.h
- camra/pfip_client.cpp
- tcs/pfip_client.cpp

9.583 PFIPClientInstrumentHandler Class Reference

Public Member Functions

- **PFIPClientInstrumentHandler** (TCSDriver *d, PFIPClient &pfip, Instrument &instrument, const MessageHandlerDescriptor &m)

Public Attributes

- TCSDriver & **driver_**
- PFIPClient & **pfip_**
- Instrument & **instrument_**

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

- tcs/pfip_client.h

9.584 PFIPConfig Class Reference

A class to handle the PFIP Configuration file and settings, extending the base TCSConfig class.

9.584.1 Detailed Description

The PFIP configuration variables are defined in a static array, which must be updated if new variables are added or existing variable types are changed.

All the PFIP configuration options are held inside the object in a map data structure. There are a variety of "get" operations to get the option values by name and type.

`#include <PFIPConfig.h>`

Public Member Functions

- PFIPConfig (bool isNamed=0)

  Constructor with base initializer for TCSConfig.

- virtual ~PFIPConfig ()

  Destructor.

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

- virtual void print_summary ()

  Called prior to displaying options.

- int **GetStringArrayIndex** (const string names[ ], int maxNames, const string &name)
- string **GetStringArrayName** (const string names[ ], int maxNames, int index)
- string **GetStringArrayNames** (const string names[ ], int maxNames)
- fptime **GetEPICSTimeout** ()

Public Attributes

- bool **m_HardwareDisabled**
- bool **m_FCUDisabled**
- bool **m_PFIPMonitorDisabled**
- bool **m_PFIPDisabled**
- bool **m_VEncl1Disabled**
- bool **m_VEncl2Disabled**
- bool **m_ENWAPositionSensorDisabled**
- bool **m_ProbeDisabled** [pfip::NUMBER_OF_PROBES]
- string **m_ProbeDisabledStr** [pfip::NUMBER_OF_PROBES+1]
- bool **m_EPICSGetsDebug**
- bool **m_EPICSSetsDebug**
- bool **m_EPICSSummaryDebug**
- bool **m_ProbeMotorAndEncoderLoggingEnabled**
- bool **m_ProbeMotorAndEncoderEventsEnabled**
- bool **m_ProbeErrorStatusBitsLoggingEnabled**

- bool **m_ProbeAllStatusBitsLoggingEnabled**
- string **m_EPICS_ip_address**
- string **m_FCU_ip_address**
- int **m_FCU_port**
- string **m_PFIPEncl_ip_address**
- int **m_PFIPEncl_port**
- string **m_VEncl1_ip_address**
- int **m_VEncl1_port**
- string **m_VEncl2_ip_address**
- int **m_VEncl2_port**
- float **m_FCUStatusLoopFrequency_sec**
- float **m_FCUStatusEventFrequency_sec**
- float **m_ACQStatusLoopFrequency_sec**
- float **m_ACQStatusEventFrequency_sec**
- float **m_PFIPMonitorStatusLoopFrequency_sec**
- float **m_VirusMonitorStatusLoopFrequency_sec**
- float **m_ACQFocusScaleSlope**
- float **m_ACQFocusScaleConstant**
- float **m_DewPointScaleSlope**
- float **m_DewPointScaleConstant**
- float **m_FPAPurgeAirFlowScaleSlope**
- float **m_FPAPurgeAirFlowScaleConstant**
- float **m_WFCPurgeGN2FlowScaleSlope**
- float **m_WFCPurgeGN2FlowScaleConstant**
- float **m_VENCPurgeAirFlowScaleSlope**
- float **m_VENCPurgeAirFlowScaleConstant**
- float **m_VENCPressureScaleSlope**
- float **m_VENCPressureScaleConstant**
- float **m_HEPAPressureScaleSlope**
- float **m_HEPAPressureScaleConstant**
- float **m_PFIPTimeout**
- float **m_FCUTimeout**
- float **m_ProbeMoveTimeout**
- float **m_FCUHeadMoveTimeout**
- float **m_ACQMirrorMoveTimeout**
- float **m_EPICSTimeout**
- int **m_ExposureDelaySec**
- int **m_ShutterOpenEncoderPos**
- int **m_ShutterClosedEncoderPos**
- int **m_ShutterEncoderMask**
- int **m_ShutterAccuracy**
- int **m_ENWAFilter1EncoderPos**
- int **m_ENWAFilter2EncoderPos**
- int **m_ENWAFilter3EncoderPos**
- float **m_ProbeProximityTolerance_radians**
- int **m_ProbeEncoderMoveAccuracy**
- int **m_ArmEncoderMoveAccuracy**
- int **m_ProbeInitialIGains** [pfip::NUMBER_OF_PROBES]
- int **m_ProbeInitialPGains** [pfip::NUMBER_OF_PROBES]
- bool **m_DisableProbeMoveSettlingGains**

- int **m_ProbeMoveSettlingIGains** [pfip::NUMBER_OF_PROBES]
- int **m_ProbeMoveSettlingPGains** [pfip::NUMBER_OF_PROBES]
- float **m_ProbeMoveCheckOnTargetFrequency_sec**
- int **m_ProbeChangeGainsAt** [pfip::NUMBER_OF_PROBES]
- bool **m_DisableProbeSmallMoveGains**
- int **m_ProbeSmallMoveIGains** [pfip::NUMBER_OF_PROBES]
- int **m_ProbeSmallMovePGains** [pfip::NUMBER_OF_PROBES]
- int **m_ProbeUseSmallGainsAt** [pfip::NUMBER_OF_PROBES]
- string **m_Guider1CameraFilters** [5]
- string **m_Guider2CameraFilters** [5]
- string **m_ACQCameraFilters** [4]
- string **m_FCUSliderALamps** [5]
- string **m_FCUSliderBLamps** [5]
- string **m_FCUSliderALampIDs** [5]
- string **m_FCUSliderBLampIDs** [5]
- string **m_FCULDLSLampID**
- string **m_FCULightBanks** [3]
- string **m_FCULightGuides** [2]
- string **m_ENWAFilters** [3]
- string **m_EXWAFilters** [2]

9.584.2 Member Function Documentation

9.584.2.1 virtual void PFIPConfig::print_summary () [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.584.2.2 void PFIPConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- pfip/include/PFIPConfig.h
- pfip/src/PFIPConfig.cpp

9.585 Ui::pfipControlTab Class Reference

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

- gui/build/ui_pfipControlTab.h

9.586 pfipControlTabUi Class Reference

Signals

- void **IssuePFIPControl** (QString, QString, bool)
- void **SendPFIPCommand** (const std::string, Message ∗)
- void **SetACQCameraFilter** (std::string)
- void **SetACQCameraFocusRel** (double)
- void **SetACQCameraFocusAbs** (double)
- void **SetDitherPosition** (int)
- void **DeployACQCameraMirror** (void)
- void **RetractACQCameraMirror** (void)
- void **DeployPV_CWFSCameraMirror** (void)
- void **RetractPV_CWFSCameraMirror** (void)
- void **DeployPVCamera** (void)
- void **DeployCWFSCamera** (void)

Public Member Functions

- **pfipControlTabUi** (const char ∗name, QWidget ∗pLoadedWidget, guiConfig ∗pConfig)

Protected Attributes

- QString **m_filterNamesACQ**
- QString **m_filterNamesG1**
- QString **m_filterNamesG2**

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

- gui/include/pfipControlTabUi.h
- gui/build/moc_pfipControlTabUi.cpp
- gui/src/pfipControlTabUi.cpp

9.587 pasImagePipelineSubSystem::PFIPData Struct Reference

Public Attributes

- tthread::mutex **accessMutex**
- string **cameraFilterName**
- int **cameraFilterIndex**
- string **FCULampAName**
- string **FCULampBName**
- bool **bFCULampAState**
- bool **bFCULampBState**
- bool **bFCULDLSState**
- string **FCUInputLightBankName**
- string **FCUOutputLightGuideName**
- string **FCUHeadPositionName**
- double **FCUExposureSecs**
- string **FCUExposingLamp**
- string **FCUSliderName**
- string **FPAShutterPositionName**
- int **ENWAFilterPosition**
- string **ENWAFilterName**
- int **EXWAFilterPosition**
- string **EXWAFilterName**
- int **DitherPosition**
- string **FCULamp1AId**
- string **FCULamp2AId**
- string **FCULamp3AId**
- string **FCULamp4AId**
- string **FCULamp5AId**
- string **FCULamp1BId**
- string **FCULamp2BId**
- string **FCULamp3BId**
- string **FCULamp4BId**
- string **FCULamp5BId**
- string **FCULampLDLSId**

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

- pas/include/pasImagePipelineSubSystem.h

9.588 Ui::pfipEnvironmentTab Class Reference

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

- gui/build/ui_pfipEnvironmentTab.h

9.589 pfipEnvironmentTabUi Class Reference

Signals

- void **ForceHeartbeat** (std::string)

Public Member Functions

- **pfipEnvironmentTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

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

- gui/include/pfipEnvironmentTabUi.h
- gui/build/moc_pfipEnvironmentTabUi.cpp
- gui/src/pfipEnvironmentTabUi.cpp

9.590 PFIPException Class Reference

Public Member Functions

- **PFIPException** (tcsexcept::tcs_exception_t __pfip, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.591 PFIPHandler Class Reference

A class derived from Handler to encompass all the message handlers for PFIP server.

`#include <pfipServer.h>`

Public Member Functions

- PFIPHandler (const string &name, Receiver *rcv, ACQ_SubSystem *acq, MessageHandlerDescriptor &usage)

  Constructor.

- void SetupMoveProbes (const string &handlerName, MoveProbeParms *const &position)

  Initialize position structures for a move.

- pfip::ProbeID LookupProbe (const std::string &name)

  Convert probe name to an id.

- virtual void HandlerAction (Message *pMessage)=0

  Do the work requested by the handler Each handler must implement.

- void process (Message *pMessage)

  Implement the thread that runs the handler action.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

Protected Attributes

- ACQ_SubSystem * **m_pACQ**

9.591.1 Constructor & Destructor Documentation

9.591.1.1 PFIPHandler::PFIPHandler (const string & *name*, Receiver * *rcv*, ACQ_SubSystem * *acq*, MessageHandlerDescriptor & *usage*) [inline]

Parameters:

← ***name*** - a std::string containing the name of the command handler. This is the name that will be used to identify the command.

← ***rcv*** - a pointer to the receiver object associated with this handler

← ***acq*** - a pointer to the ACQ_SubSystem object that manages the PFIP acquisition system

← ***usage*** - a std::string that describes the command for the common help()

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp
- pfip/src/ProbeAssemblyHandlers.cpp

9.592 PFIPInterrupt Class Reference

9.592.1 Detailed Description

Interrupt the shutter exposure, probe move, or probe enable/disable operation in progress.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PFIPInterrupt** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.593 PFIPLampsDB Class Reference

Public Member Functions

- bool **IsDBOpen** ()
- bool **StoreLampElapsed** (int lampId, double Value)

Static Public Member Functions

- static PFIPLampsDB & **instance** ()

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

- pfip/include/PFIPLampsDB.h
- pfip/src/PFIPLampsDB.cpp

9.594 PFIPModbusException Class Reference

Public Member Functions

- **PFIPModbusException** (tcsexcept::tcs_exception_t __pfiplib, const std::string &w, const char *file, int line, const char *func)

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

- het_lib/common/include/tcs_exception.h

9.595 PFIPMonitorStatus Class Reference

Classes

- struct alarms
- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- status **m_status**
- alarms **m_alarms**

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

- gui/include/pfipStatusData.h
- gui/src/pfipStatusData.cpp

9.596 PFIPProbeInfo Class Reference

A class encapsulating hardware status for PFIP probes.

`#include <PFIPProbeInfo.h>`

Public Member Functions

- **PFIPProbeInfo** (const string &sourceName)
- bool ReadProbeStatus (EPICSComm *epicsComm, bool bLogStatusBits=0)

  Read the hardware status of all probes from EPICS variables.

- bool **ReadMAXONDriveData** (EPICSComm *epicsComm, bool bSendEvent=0)
- bool WaitForProbeEnabled (pfip::ProbeID probe, const string &probeName, EPICSComm *epicsComm, const double timeoutSeconds, volatile bool &interruptRequested)

  Wait until the probe has been enabled/disabled.

- bool **WaitForProbeDisabled** (pfip::ProbeID probe, EPICSComm *epicsComm, const double timeoutSeconds, volatile bool &interruptRequested)
- void GetDriveStatus (const pfip::ProbeID p, MAXONDriveStatus &s)

  Return a pointer to a structure containing the hardware status bits.

- void **GetDriveData** (const pfip::ProbeID p, MAXONDriveData &s)
- bool IsTargetReachedErrorOrDisabled (pfip::ProbeID p, string probeName)

  Returns true if a status update was received for the indicated probe.

- bool IsFaulted (EPICSComm *epicsComm, pfip::ProbeID p, bool bLogStatusBits=0)

  Returns true if the hardware status for the indicated probe is in a fault state.

- bool TargetReached (pfip::ProbeID p, string probeName)

  Returns true if the hardware status for the indicated probe indicates target reached.

- void LogErrors (pfip::ProbeID p)

  Returns true if the hardware status for the indicated probe indicates target reached (Arm or Probe).

- bool IsEnabled (pfip::ProbeID p)

  Return boolean indicating if the probe and arm is enabled/disabled.

- bool IsProbeEnabled (EPICSComm *epicsComm, pfip::ProbeID probe)
- bool **IsProbeDisabled** (EPICSComm *epicsComm, pfip::ProbeID probe, bool bLogStatusBits=0)

Public Attributes

- tthread::Timer **targetTimer_** [pfip::NUMBER_OF_PROBES]
- tthread::Timer **moveCompleteTimer_** [pfip::NUMBER_OF_PROBES]
- bool **targetReached_** [pfip::NUMBER_OF_PROBES]
- bool **moveComplete_** [pfip::NUMBER_OF_PROBES]
- bool **gainChanged_** [pfip::NUMBER_OF_PROBES]
- bool **smallMoveGainIsSet_** [pfip::NUMBER_OF_PROBES]
- int **moveCompleteCount_** [pfip::NUMBER_OF_PROBES]
- MAXONStatusInfo **stat_** [pfip::NUMBER_OF_PROBES]

9.596.1 Member Function Documentation

9.596.1.1 bool PFIPProbeInfo::IsProbeEnabled (EPICSComm * *epicsComm*, pfip::ProbeID *probe*)

Parameters:

← ***epicsComm*** - ptr to the EPICS communication object

← ***probe*** - probe type

← ***setting*** - if true, return true if probe is enabled; if false, return true if probe is disabled

9.596.1.2 void PFIPProbeInfo::LogErrors (pfip::ProbeID *p*)

Write log messages indicating the hardware status of the indicated probe

9.596.1.3 bool PFIPProbeInfo::WaitForProbeEnabled (pfip::ProbeID *probe*, const string & *probeName*, EPICSComm * *epicsComm*, const double *timeoutSeconds*, volatile bool & *interruptRequested*)

Parameters:

← ***probe*** - the probe identifier

← ***epicsComm*** - the EPICS communication object

← ***setting*** - if true, wait for probe to be enabled; if false, wit for probe to be disabled

← ***interruptRequested*** - if true, the wait operation was interrupted

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

- pfip/include/PFIPProbeInfo.h
- pfip/src/PFIPProbeInfo.cpp

9.597 pfipServer Class Reference

9.597.1 Detailed Description

A class to contain all the PFIP commands and start/stop their receiver. Based on the common tcs subsystem class.

`#include <pfipServer.h>`

Public Member Functions

- pfipServer (const string &appName, PFIPConfig *config)
- virtual ~pfipServer ()

  Destructor.

- bool init ()

  Allocate all the PFIP objects and handlers.

- int start ()

  Start the PFIP controller, the PFIP receiver and all the command handlers.

- void stop ()

  Stop the PFIP controller and close all the command receivers.

Public Attributes

- PFIPConfig *const **m_pConfig**

9.597.2 Constructor & Destructor Documentation

9.597.2.1 pfipServer::pfipServer (const string & *appName*, PFIPConfig * *config*)

Constructor

Parameters:

← ***ptr*** to the initialized PFIP configuration object

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

- pfip/include/pfipServer.h
- pfip/src/pfipServer.cpp

9.598 PFIPStatusActions Class Reference

Public Member Functions

- PFIPStatusActions ()
- bool **waitForEnable** ()
- bool **waitForDisable** ()

Static Public Member Functions

- static bool **waitForSettingInt** (EPICSComm *epicsComm, pfip::EPICSVariableLookupKey varKey, const int value, const fptime &timeout)
- static bool **waitForSettingBit** (EPICSComm *epicsComm, pfip::EPICSVariableLookupKey varKey, const unsigned int value, const fptime &timeout)
- static bool **waitForSettingBool** (EPICSComm *epicsComm, pfip::EPICSVariableLookupKey varKey, const bool value, const fptime &timeout)

Public Attributes

- MAXONStatusInfo **status**

9.598.1 Constructor & Destructor Documentation

9.598.1.1 PFIPStatusActions::PFIPStatusActions () `[inline]`

Constructor

Parameters:

← ***The*** PFIP configuration object

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

- pfip/include/PFIPStatus.h
- pfip/src/PFIPStatus.cpp

9.599 PFIPStatusBase Class Reference

Classes

- struct flags
- struct status

Public Member Functions

- **PFIPStatusBase** (const char ∗name)

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

- gui/include/pfipStatusData.h

9.600 liborchestrate::PfipSubsystem Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.601 tcs_photometry::PhotData Struct Reference

9.601.1 Detailed Description

result of fixed and kron aperture photometry

`#include <object.h>`

Public Member Functions

- void **setflag** (const int flag)

Public Attributes

- float **xc_**
- float **yc_**
- float **ir1_**
- float **kron_aper_**
- float **fixed_aper_**
- ApperturePhotResults **kron_p_**
- ApperturePhotResults **fixed_p_**
- int **flags_**

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

- het_lib/photometry/include/object.h

9.602 tcs_photometry::Photometry Class Reference

9.602.1 Detailed Description

Object Detection and Photometry Class

Possible usage includes:

- Fit a global background polynomial to the whole image and subtract
- Subtract the local background around our guide star using the background signal in an annulus around the star. A first-order surface is fitted to the annulus and subtracted.
- Detect sources in the rectangular region around the guide star. Sources are defined as a minimum number of adjacent pixels with flux above some factor times the background rms noise.
- For each source
- Measure the source's parameters
- Perform Kron-aperture photometry on the source
- Fit a PSF model to the source

For examples of usage see testphot.cpp

```
#include <photometry.h>
```

Public Types

- typedef ltl::MArray< float, 2 > **MImage**
- typedef ltl::MArray< char, 2 > **MMask**
- typedef std::list< Object * > **ObjectList**

Public Member Functions

- **Photometry** (const ConfigureParams &configParams)
- void **getOptions** (ConfigureParams &configParams)
- void **resetOptions** (const ConfigureParams &configParams)
- void **setZeropoint** (double zeroPoint, double exposureTimeSec, double illum_frac)
- void setGuideRect (const GuideRect &a)
- void resetGuideRect ()
- void setPixscale (const double p)
- void **setResid** (const bool r)
- GuiderMetrologyData Guider_photometry (MImage &I, std::ostream *os=NULL)
- TTCAMMetrologyData TTCAM_photometry (MImage &A, std::ostream *os=NULL)
- MImage copy_square (const MImage &A, const int minx, const int maxx, const int miny, const int maxy)
- int subtract_background_fit (MImage &I)
- int subtract_background_fit_annulus (MImage &I)
- int subtract_background_fit_outer_edge (MImage &I)
- ObjectList find_stars (const MImage &I)
- int kron_aperture_phot (const MImage &I, const Object &o, PhotData &pd, ApperturePhotResults &res)

- int fixed_aperture_phot (const MImage &I, const Object &o, PhotData &pd, ApperturePhotResults &res)
- int fast_kron_aperture_phot (const MImage &I, const Object &o, PhotData &pd, ApperturePhotResults &res)
- int fast_fixed_aperture_phot (const MImage &I, const Object &o, PhotData &pd, ApperturePhotResults &res)
- int fit_bivariate_gaussian (MImage &A, const float x, const float y, const int radius, FitPars &pd, bool resid=false)
- int fit_bivariate_moffat (MImage &A, const float x, const float y, const int radius, FitPars &fp, bool resid=false)
- void **delete_objectlist** (Photometry::ObjectList &l)

Protected Member Functions

- int aperture_phot (const MImage &Image, const float x, const float y, const float aper, const float sky1, const float sky2, PhotData &pd, ApperturePhotResults &res)
- int fast_aperture_phot (const MImage &Image, const float x, const float y, const float aper, PhotData &pd, ApperturePhotResults &res)
- std::list< Object * > assemble (MMask &M, const MImage &I)
- void mark_pixels (Object &obj, const int x, const int y, MMask &M, const MImage &I)
- float integrate_soft_e_mask_fixed (const MImage &I, const float sky, float xc, float yc, float ap, const int subsample, float *area)
- Photometry::MImage elliptical_mask (const MImage &I, const float xc, const float yc, const float a1, const float b1, const float pa)
- Photometry::MImage elliptical_annulus_mask (const MImage &I, const float xc, const float yc, const float a1, const float b1, const float a2, const float b2, const float pa)
- Photometry::MImage square_annulus_mask (const MImage &I, const float x1, const float y1, const float x2, const float y2)
- bool check_variance (const MImage &I)

Protected Attributes

- ConfigureParams pp_

  configuration parameter block for the photometry routines

- GuideRect analysis_region_

  the image region of interest

- double **bg_**
- double sigma_bg_

  background value and stddev.

- double residual_bg_

  residual local background for fit

- double pixscale_

  pixel scale if we know it

- ltl::MArray< float, 2 > bg_image_

  the fitted background image (need to store until we know where the star is)

- bool resid_

  output residual images after PSF fitting step

9.602.2 Member Function Documentation

9.602.2.1 int tcs_photometry::Photometry::aperture_phot (const MImage & *Image*, const float *x*, const float *y*, const float *aper*, const float *sky1*, const float *sky2*, PhotData & *pd*, AperturePhotResults & *res*) [protected]

Circular aperture photometry Object center: x,y Aperture radius in pixels: aper Beginning and end of sky annulus: sky1, sky2

9.602.2.2 Photometry::ObjectList tcs_photometry::Photometry::assemble (MMask & *M*, const MImage & *I*) [protected]

segment the image into consecutive regions above threshold

9.602.2.3 bool tcs_photometry::Photometry::check_variance (const MImage & *I*) [protected]

make sure the image looks OK by looking at variances ...

9.602.2.4 MImage tcs_photometry::Photometry::copy_square (const MImage & *A*, const int *minx*, const int *maxx*, const int *miny*, const int *maxy*)

Copy out a rectangular region out of an image. The new image retains the index ranges corresponding to the coordinates within the original image.

Copy out a square region containing the circle centered at x,y with radius. The new image retains the index ranges corresponding to the coordinates within the original image.

9.602.2.5 Photometry::MImage tcs_photometry::Photometry::elliptical_annulus_mask (const MImage & *I*, const float *xc*, const float *yc*, const float *a1*, const float *b1*, const float *a2*, const float *b2*, const float *pa*) [protected]

Return an image multiplied by the characteristic function of an elliptical annulus. All pixels outside the annulus will be set to zero. Pixels within the annulus will retain their values.

9.602.2.6 Photometry::MImage tcs_photometry::Photometry::elliptical_mask (const MImage & *I*, const float *xc*, const float *yc*, const float *a1*, const float *b1*, const float *pa*) [protected]

Return an image multiplied by the characteristic function of an ellipse All pixels outside the ellipse will be set to zero, pixels within the ellipse will retain their values.

9.602.2.7 int tcs_photometry::Photometry::fast_aperture_phot (const MImage & *Image*, const float *x*, const float *y*, const float *aper*, PhotData & *pd*, ApperturePhotResults & *res*) [protected]

Circular aperture photometry. Faster version with no background subtraction. Used in guider mode where the background has already been processed on the guide rectangle. Object center: x,y Aperture radius in pixels: aper

9.602.2.8 int tcs_photometry::Photometry::fast_fixed_aperture_phot (const MImage & *I*, const Object & *o*, PhotData & *pd*, ApperturePhotResults & *res*) [inline]

Photometry in fixed-sized aperture. Faster version with no background subtraction. Used in guider mode where the background has already been processed on the guide rectangle.

9.602.2.9 int tcs_photometry::Photometry::fast_kron_aperture_phot (const MImage & *I*, const Object & *o*, PhotData & *pd*, ApperturePhotResults & *res*) [inline]

Photometry in Kron-like aperture (aperture size based on a constant times the intensity-weighted first moment of the image). Faster version with no background subtraction. Used in guider mode where the background has already been processed on the guide rectangle.

9.602.2.10 Photometry::ObjectList tcs_photometry::Photometry::find_stars (const MImage & *I*)

threshold the image at some level above the background noise, segment the resulting image mask and assemble `Objects` out of consecutive regions.

9.602.2.11 int tcs_photometry::Photometry::fit_bivariate_gaussian (MImage & *A*, const float *x*, const float *y*, const int *radius*, FitPars & *fp*, bool *resid* = false)

Fit a bivariate Gaussian ellipsoid to the image of a source

9.602.2.12 int tcs_photometry::Photometry::fit_bivariate_moffat (MImage & *A*, const float *x*, const float *y*, const int *radius*, FitPars & *fp*, bool *resid* = false)

Fit a bivariate Moffat to the image of a source I(x,y) = I0 / [1 + (x/X)^2 + (y/Y)^2]^beta

9.602.2.13 int tcs_photometry::Photometry::fixed_aperture_phot (const MImage & *I*, const Object & *o*, PhotData & *pd*, ApperturePhotResults & *res*) [inline]

Photometry in fixed-sized aperture

9.602.2.14 GuiderMetrologyData tcs_photometry::Photometry::Guider_photometry (MImage & *I*, std::ostream * *os* = NULL)

Detection and kron,aperture,psf-fitting photometry of a single source within the guide rectangle after background subtraction.

9.602.2.15 float tcs_photometry::Photometry::integrate_soft_e_mask_fixed (const MImage & *I*, const float *sky*, float *xc*, float *yc*, float *ap*, const int *subsample*, float * *area*) [protected]

Integrate the flux within an elliptical aperture by subsampling the image and computing the intersection of the ellipse with the subsampled pixels.

9.602.2.16 int tcs_photometry::Photometry::kron_aperture_phot (const MImage & *I*, const Object & *o*, PhotData & *pd*, ApperturePhotResults & *res*) [inline]

Photometry in Kron-like aperture (aperture size based on a constant times the intensity-weighted first moment of the image)

9.602.2.17 void tcs_photometry::Photometry::mark_pixels (Object & *obj*, const int *x*, const int *y*, MMask & *M*, const MImage & *I*) [protected]

recursively add pixels surrounding a pixel (x,y) to an object if they are flagged in the mask as lying above threshold

9.602.2.18 void tcs_photometry::Photometry::resetGuideRect () [inline]

Reset the analysis region to the full frame

9.602.2.19 void tcs_photometry::Photometry::setGuideRect (const GuideRect & *a*) [inline]

Set the region of interest. Object detection and photometry will be restricted to this region. Initia background subtraction will still consider the full image. In guider mode, this is the guide rectangle

9.602.2.20 void tcs_photometry::Photometry::setPixscale (const double *p*) [inline]

set a pixelscale for sky mag in mags/arcsec^2

9.602.2.21 Photometry::MImage tcs_photometry::Photometry::square_annulus_mask (const MImage & *I*, const float *x1*, const float *y1*, const float *x2*, const float *y2*) [protected]

Return an image multiplied by the characteristic function of an square annulus. All pixels outside the annulus will be set to zero. Pixels within the annulus will retain their values.

Return an image multiplied by the characteristic function of an square annulus around the image. All pixels outside the annulus will be set to zero. Pixels within the annulus will retain their values.

9.602.2.22 int tcs_photometry::Photometry::subtract_background_fit (MImage & *I*)

subtract the background from an image using a k-th order surface fit to the image values. Also computes the RMS within the image.

9.602.2.23 int tcs_photometry::Photometry::subtract_background_fit_annulus (MImage & *I*)

subtract the background from an image using a 2nd order surface fit to the image values in an annulus. Also computes the RMS within that annulus.

subtract the background from an image using a first-order surface fit to the image values in an annulus. Also computes the RMS within that annulus.

9.602.2.24 int tcs_photometry::Photometry::subtract_background_fit_outer_edge (MImage & *I*)

subtract the background from an image using a first-order surface fit to the image values within sky_annulus pixels around the image borders

9.602.2.25 TTCAMMetrologyData tcs_photometry::Photometry::TTCAM_photometry (MImage & *A*, std::ostream ∗ *os* = `NULL`)

Detection and centroiding/shape measurements of a single source within the image. No background subtraction.

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

- het_lib/photometry/include/photometry.h
- het_lib/photometry/src/photometry.cpp

9.603 PhotometryData Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual void **newData** (int size)
- virtual void **deleteData** (void)

Public Attributes

- ulong **Nelem**
- double **nsigma**
- double * **bgr**
- double * **ron**
- double * **flux**
- double * **snr**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.604 Ui::photometryOptionsDialog Class Reference

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

- gui/build/ui_photometryOptionsDlg.h

9.605 photometryOptionsDialog Class Reference

Public Member Functions

- **photometryOptionsDialog** (const string &pipeline, const string &type, QWidget *parent=0)
- void **loadData** (const tcs_photometry::ConfigureParams &options)
- void **getData** (tcs_photometry::ConfigureParams &options)

Protected Member Functions

- void **loadSettings** (void)
- void **saveSettings** (void)

Protected Attributes

- tcs_photometry::ConfigureParams **options**

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

- gui/include/photometryOptionsDlg.h
- gui/src/photometryOptionsDlg.cpp

9.606 PIDHandler< T > Class Template Reference

template<typename T> class PIDHandler< T >

Public Member Functions

- **PIDHandler** (MetrologyLoop *l, MetrologyPIDLoop< T > *pid, const MessageHandlerDescriptor &m)

Protected Attributes

- MetrologyPIDLoop< T > * **pid_**

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

- tcs/metrology.h

9.607 PIDLoop< M, distance > Class Template Reference

9.607.1 Detailed Description

template<typename M, typename distance = std::minus<M>> class PIDLoop< M, distance >

notional sketch of a generic PID loop

Expects a data type M, and a distance measure of the form M distance(M,M). The distance measure defaults to operator-(M,M). It is further assumed that M supports basic arithmetic operators +,-,*,/ with other objects of type M and double.

`#include <tcs_pid.h>`

Public Member Functions

- **PIDLoop** (double kp, double ki, double kd)
- void **apply_setpoint** (const M &setpoint)
- void load_measurement (const M &measurement)
- void **load_error** (const M &error)
- const M & **get_setpoint** () const
- const M & **get_error** () const
- const bool **setpoint_applied** () const
- double **get_kp** () const
- double **get_ki** () const
- double **get_kd** () const
- void **reset** ()

- void set_proportional (const double kp)
- void **set_integral** (const double ki)
- void **set_derivative** (const double kd)
- void **set_period** (const double seconds)

Protected Attributes

- bool **setpoint_applied_**
- M **error_**
- M **prior_error_**
- M **setpoint_**
- M **integral_**
- M **derivative_**
- M **output_error_**
- double **kp_**
- double **ki_**
- double **kd_**
- double **dt_**

9.607.2 Member Function Documentation

9.607.2.1 template<typename M, typename distance = std::minus<M>> void PIDLoop< M, distance >::load_measurement (const M & *measurement*) [inline]

Load a new measurement into the loop. The distance function supplied as a template argument will be used to calculate the current error.

9.607.2.2 template<typename M, typename distance = std::minus<M>> void PIDLoop< M, distance >::set_proportional (const double *kp*) [inline]

Set the PID loop constants

Classic Ziegler-Nichols Kp = 0.6 Ku Ti = 0.5 Tu Td = 0.125 Tu Pessen Integral Rule Kp = 0.7 Ku Ti = 0.4 Tu Td = 0.15 Tu Some Overshoot Kp = 0.33 Ku Ti = 0.5 Tu Td = 0.33 Tu No Overshoot Kp = 0.2 Ku Ti = 0.5 Tu Td = 0.33 Tu

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

- het_lib/common/include/tcs_pid.h

9.608 pingHandler Class Reference

Public Member Functions

- **pingHandler** (int limit)

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

- playground/pong/pong.cpp

9.609 PipelineConfig Class Reference

Public Member Functions

- void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- double m_pipelineCycleTime_secs

  Pipeline cycle time (time between exposures).

- int m_imagesPerCycle

  Images to co-add per pipeline cycle).

- string m_lightSource

  Optional light source (config file will be <lightSource>Light.conf).

- LightSourceConfig * m_pLightSourceConfig

  Light source configuration.

- string m_camera

  Camera (config file will be <camera>Camera.conf).

- CameraConfig * m_pCameraConfig

  Camera configuration.

- string m_device

  Device (config file will be <device>Device.conf).

- DeviceConfig * m_pDeviceConfig

  Device configuration.

- string m_cleaner

  Optional image cleaner (config file will be <cleaner>Cleaner.conf).

- bool m_bCleaningEnabled

  Image cleaning enabled.

- CleanerConfig * m_pCleanerConfig

  Cleaner configuration.

- string m_processor

  Optional image processor (config file will be <processor>Processor.conf).

- bool m_bProcessingEnabled

  Image processing enabled.

- ProcessorConfig * m_pProcessorConfig

  Processor configuration.

- string m_storage

  Optional storage module.

- bool m_bStoringEnabled

  Image storing enabled.

- bool m_bPreProcessorStoringEnabled

  Store image before processing enabled.

- string m_storageDirectory

  Directory for storing image files.

9.609.1 Member Function Documentation

9.609.1.1 void PipelineConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

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

- pas/include/pasConfig.h
- pas/src/pasConfig.cpp

9.610 PipelineCyclePipelineHandlerClass Class Reference

Public Member Functions

- **PipelineCyclePipelineHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.610.1 Member Function Documentation

9.610.1.1 virtual void PipelineCyclePipelineHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasPipelineSubSystem.h

9.611 PipelineOnLine Class Reference

Check to see if pipeline is running.

9.611.1 Detailed Description

Parameters:

none

`#include <pasPipelineSubSystem.h>`

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

- pas/include/pasPipelineSubSystem.h

9.612 PipelineRelinquishControlHandlerClass Class Reference

Public Member Functions

- **PipelineRelinquishControlHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.612.1 Member Function Documentation

9.612.1.1 virtual void PipelineRelinquishControlHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasPipelineSubSystem.h

9.613 PipelineSetCycleTimeHandlerClass Class Reference

Public Member Functions

- **PipelineSetCycleTimeHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.613.1 Member Function Documentation

9.613.1.1 virtual void PipelineSetCycleTimeHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasPipelineSubSystem.h

9.614 PipelineSetIntervalSpeedHandlerClass Class Reference

Public Member Functions

- **PipelineSetIntervalSpeedHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.614.1 Member Function Documentation

9.614.1.1 virtual void PipelineSetIntervalSpeedHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasPipelineSubSystem.h

9.615 PipelineStartPipelineHandlerClass Class Reference

Public Member Functions

- **PipelineStartPipelineHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.615.1 Member Function Documentation

9.615.1.1 virtual void PipelineStartPipelineHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasPipelineSubSystem.h

9.616 PipelineStatus Class Reference

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus)

Public Attributes

- struct PipelineStatus::status **m_status**

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

- gui/include/pipelineStatusData.h
- gui/src/pipelineStatusData.cpp

9.617 PipelineTakeControlHandlerClass Class Reference

Public Member Functions

- **PipelineTakeControlHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.617.1 Member Function Documentation

9.617.1.1 virtual void PipelineTakeControlHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

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

- pas/include/pasPipelineSubSystem.h

9.618 tcs_photometry::Pixel Struct Reference

9.618.1 Detailed Description

Image pixel

`#include <object.h>`

Public Member Functions

- **Pixel** (const int x, const int y, const float v)
- bool operator< (const Pixel &other) const

  compare pixels by comparing their values

Public Attributes

- int **x_**
- int **y_**
- float **v_**

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

- het_lib/photometry/include/object.h

9.619 Pixel Class Reference

BGR pixel class.

Public Member Functions

- **Pixel** (unsigned char red, unsigned char green, unsigned char blue)

Public Attributes

- unsigned char **b**
- unsigned char **g**
- unsigned char **r**

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

- het_lib/tthread/test/fractal.cpp

9.620 Pixel_t Class Reference

Public Member Functions

- **Pixel_t** (const unsigned int xValue, const unsigned int yValue, const float val=0.0)

Public Attributes

- int **x**
- int **y**
- float **value**

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

- pas/include/pasImageCleanerStandard.h

9.621 ExposureInfo::PlateScale Struct Reference

Public Attributes

- double **x**
- double **y**

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

- pas/include/Camera.h

9.622 plotEventsTabUi Class Reference

Public Member Functions

- **plotEventsTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Protected Member Functions

- void **loadSettings** (void)
- void **saveSettings** (void)
- void **buildDataBasePath** (void)

Protected Attributes

- QDateTime **m_startDateTime**
- QDateTime **m_stopDateTime**
- QString **m_dataBasePath**
- QString **m_plotCommand**

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

- utility_gui/include/plotEventsTabUi.h
- utility_gui/src/plotEventsTabUi.cpp

9.623 PlotterConfig Class Reference

Public Member Functions

- **PlotterConfig** (bool isNamed=0)

Public Attributes

- int **scatter**
- double **dynamic_range**
- string **endpoint_system**
- bool **write_to_stdout**
- string **system**
- string **source**
- string **key**
- string **x**
- string **y**
- bool **x_time**

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

- plotter/tcs_plotter_config.h

9.624 PlotterDriver Class Reference

Public Member Functions

- **PlotterDriver** (const string &appName, PlotterConfig *config)
- int **start** ()
- void **start_receiver** ()
- void process (Message *m)

9.624.1 Member Function Documentation

9.624.1.1 void PlotterDriver::process (Message * *message*) [virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- plotter/tcs_plotter_driver.h
- plotter/tcs_plotter_driver.cpp

9.625 liborchestrate::PlotterTcsSubsystem Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.626 liborchestrate::PlotterTrackerSubsystem Class Reference

Public Member Functions

- def **__init__**

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

- integration/liborchestrate.py

9.627 Point Class Reference

9.627.1 Detailed Description

Main class of the library.

Contains all variables and methods for trajectory computations. Note: this is an expansion of the class 'ShotTraj' from cure/plan.

`#include <point.h>`

Public Member Functions

- Point (const MountModel &mntmod, const double RA, const double DEC, const fptime &UTC, const weather &CW, const TrackDir td, const double equinox)

  Constructors.

- virtual ~Point ()

  Destructor.

- int interp_ephem (const double MJD, double *ra, double *dec)
- Point **recalculate** (double RA, double DEC, double AZ, double EL, const pointing_coordinate &p, const weather &current_weather)
- eq_coords **get_commanded_pointing** (void)
- eq_coords **get_apparent_pointing** (void)
- double **get_parallactic_angle** (void)
- track_type_t **type** (void)
- void **update** (const double RA, const double DEC, const weather &CW, const TrackDir td)
- void **sidtraj** (double MJD, const weather &current_weather)
- double iterate (double app_ra, double app_dec, double date, double dUT1, double w, double p, double height, double x_polar, double y_polar, double temp_k, double press_m, double rhum, double comp, double UTC, double obs_ha, double *obs_dec)
- double optAz ()

  Computes and returns the optimal azimuth for given Dec.

- double UT1toUTC (const double MJD, const double dUT1)

  Convert the input MJD from UT1 to UTC.

- **Point** (const fptime &time, const std::string &object, const weather &weather, const double longitude, const double latitude, double height)
- Point (const fptime &UTC, const std::string &trajfile)

  Reads in an already computed trajectory.

- void **init** (void)
- void **set_ra_dec** (const double RA, const double DEC)
- void **set_equinox** (const double equinox)
- void **set_az_el** (const double azimuth, const double elevation)
- void **set_direction** (const TrackDir td)
- void **set_ephemeris_file** (const std::string &ephemeris_file)
- void **set_x_y** (const double x, const double y)
- void **set_satellite_file** (const std::string &satellite_file)

- void **set_weather** (const weather &weather)
- void set_rho_derivation (const rho_derivation &rd)

  Prints entire trajectory with header.

- void printTrajectory (const std::string filename, const bool full_header=1)

  Prints entire trajectory to file with optional header.

- void printInversePointing (const std::string filename, const bool full_header, const weather &weather)
- void printPoint (FILE *fp, const ShotPoint sp)

  Prints the tracker variables for a single trajectory point to a stream.

- void printEL (const std::string filename)

  Prints entire Ephemeris list.

- int **interp_ephem** (const double MJD, double *ra, double *dec)
- Point **recalculate** (double RA, double DEC, double AZ, double EL, const weather &weather)
- eq_coords **get_commanded_pointing** (void)
- eq_coords **get_apparent_pointing** (void)
- double **get_parallactic_angle** (void)
- track_type_t **type** (void)

Public Attributes

- delta_time **delta_UT_**
- fptime **time_**
- eq_coords cmd_orig_

  Original command reference coordinates for this trajectory.

- eq_coords app_orig_

  Original precessed command reference coordinates for this trajectory.

- eq_coords cmd_

  User supplied (commanded) mean ra dec equinox.

- eq_coords obs_

  Observed coordinates.

- eq_coords app_

  Apparent coordinates.

- double ha_start_

  Hour angle of the start of the trajectory.

- double ha_end_

  .....end.....

- double Az_

  Azimuth, not necessarily optimal.

- double **PA_**
- int tracks_

  Parallactic angle center track, for the requested azimuth and elevation.

- double **aoprms_** [14]
- double **MJD**
- double **date**
- double **traj_length**
- double **traj_az**
- double **sid_start**
- fptime **ut_start**
- fptime **ut_transit**
- TrackDir td_

  East or West track.

- double wave_mic_

  Parallactic angle center track, for the requested azimuth and elevation.

- double **lapse_**
- double **elevation_**
- double azimuth_

  Azimuth, not necessarily optimal.

- double optaz_

  Optimal azimuth, not necessarily pointing azimuth.

- double **parallactic_**
- TrackDir **direction_**
- std::list< ShotPoint > trajectory_

  One for each second along a track.

- std::map< std::string, std::string > **sparse_trajectory_strings_**
- int **sparse_trajectory_count_**
- fptime **sparse_trajectory_end_time_**
- std::string **object_**

9.627.2 Constructor & Destructor Documentation

9.627.2.1 Point::Point (const MountModel & *mntmod*, const double *RA*, const double *DEC*, const fptime & *UTC*, const weather & *CW*, const TrackDir *td*, const double *equinox* = 2000.0)

Sideral track, will calculate optimal Az.

9.627.3 Member Function Documentation

9.627.3.1 int Point::interp_ephem (const double *MJD*, double * *ra*, double * *dec*)

Method to interpolate between ephemeris points. Based on code by M.E. Cornell

--ORIGINAL DOCUMENTATION STARTS HERE--

interp_ephem() - interpolate between ephemeris points

Given arrays of time, RA, and Dec, use 3-point interpolation to find the coordinates at the specified input time. Arrays must be ordered in ascending time.

Arguments (all doubles): Input MJD (UTC) in days for which the coordinates are to be derived Pointer to output Right Ascension in radians Pointer to output Declination in radians

Note:

The interpolation algorithm requires 3 data points bounding the input MJD, and with valid data. The data arrays have nrecs records, with an index from 0 to nrecs-1. I'm not sure the range checking in eph_interp() is correct.

28DEC10 Written by M. E. Cornell, based on eph_interp(). 14JAN11 Note that in this routine MJD is UTC everywhere.

--ORIGINAL DOCUMENTATION ENDS HERE--

AUG 2013 Integration into point library by Martin Landriau Static global variables replaced by point class members. Messages now handled by TCS logger.

9.627.3.2 double Point::iterate (double *app_ra*, double *app_dec*, double *date*, double *dUT1*, double *w*, double *p*, double *height*, double *x_polar*, double *y_polar*, double *temp_k*, double *press_m*, double *rhum*, double *comp*, double *UTC*, double *obs_ha*, double * *obs_dec*)

Iterate, using slalib, to find crossing, start and stop times. This algorithm is discussed in HET Technical Report #90, eqns 10.1 - 10.10. It is only called by sidtraj().

9.627.3.3 double Point::optAz ()

The algorithm for best (east) azimuth is taken directly from tcsmon/rdscope.c Note that we use the apparent declination, rather than the observed

9.627.3.4 void Point::printEL (const std::string *filename*)

Prints a satelitte's computed position on sky on requested night.

9.627.3.5 void Point::printInversePointing (const std::string *filename*, const bool *full_header*, const weather & *weather*)

Prints the computed apparent RA/DEC for each point along a track to test consistency of forward and backward coordinate transforms

9.627.4 Member Data Documentation

9.627.4.1 int Point::tracks_

Number of available tracks (2-> E+W tracks distinct; 1-> E+W tracks merged; 0-> unobservable).

Number of available tracks (2-> E+W tracks distinct; 1-> E+W tracks merged; 0-> unobservable)

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

- integration/sir/Het_trajectory/point.h
- tcs/traj_comp/point.h
- integration/sir/Het_trajectory/point.cpp
- integration/sir/Het_trajectory/sidtraj.cpp
- tcs/traj_comp/nsotraj.cpp
- tcs/traj_comp/point.cpp
- tcs/traj_comp/sattraj.cpp
- tcs/traj_comp/sidtraj.cpp
- tcs/traj_comp/xytraj.cpp

9.628 point Struct Reference

Public Attributes

- double **enc_t**
- double **enc_p**
- double **x**
- double **y**
- double **z**

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

- het_lib/coord_trans/mount_model/encoder_fit.cpp

9.629 Point_t Class Reference

Public Member Functions

- **Point_t** (int xValue, int yValue)

Public Attributes

- int **x**
- int **y**

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

- pas/include/pasImageCleanerStandard.h

9.630 pointing_coordinate Class Reference

Public Member Functions

- **pointing_coordinate** (double ra, double dec, double az, double el, double lst, itf_coordinate itf)
- virtual void **apply_parms** (Payload &p) const

Public Attributes

- double **ra**
- double **dec**
- double **az**
- double **el**
- double **lst**
- double **pa**
- double **ha**
- double zd

  apparent coordinates, current epoch

- double **ra_2000**
- double dec_2000

  mean J2000 coordinates, used only for pointing data

- itf_coordinate **itf**

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

- het_lib/common/include/tcs_coordinates.h
- het_lib/common/src/tcs_coordinates.cpp

9.631 pointing_ForceStatusEvents Class Reference

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

- pointing/include/pointing_handler_doc.h

9.632 pointingConfig Class Reference

Public Member Functions

- **pointingConfig** (bool isNamed=0)
- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

- virtual void print_summary (void)

  Called prior to displaying options.

Public Attributes

- string **pointing_route**
- string **pointing_event_route**

9.632.1 Member Function Documentation

9.632.1.1 virtual void pointingConfig::print_summary (void) [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.632.1.2 void pointingConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- pointing/include/pointingConfig.h
- pointing/src/pointingConfig.cpp

9.633 pointingServer Class Reference

Public Member Functions

- **pointingServer** (const string &appName, pointingConfig *pConfig)
- virtual int **start** (void)
- virtual void **stop** (void)
- **SERVER_HANDLER_DECL** (ForceStatusEvents)

Protected Member Functions

- void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- void **SetName** (string name)
- const char * **GetName** (void)
- void **SetDebug** (bool on)

Protected Attributes

- string **m_Name**
- bool **m_Debug**
- pointingConfig * **m_Config**
- Receiver * **m_Receiver**

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

- pointing/include/pointingServer.h
- pointing/src/pointingServer.cpp

9.634 Pool< T > Class Template Reference

9.634.1 Detailed Description

template<typename T> class Pool< T >

Memory pool for allocations of elements of a single fixed size. Memory is allocated from the OS in chunks, the pool grows in capacity as elements are allocated from it. The pool does not shrink.

The pool can be used directly by calling its API, or using overloaded `operator new` and `delete` as follows:

```
struct foo { void *operator new (size_t size); void operator delete
(void *ptr); };

Pool<foo> fpool (16);

void* foo::operator new (size_t size) { return fpool.Allocate (); }

void foo::operator delete (void *ptr) { fpool.Deallocate (ptr); }

#include <pool.h>
```

Public Member Functions

- Pool (size_t block_size)
- ~Pool ()

  Checks for emptiness before destructing.

- void * Allocate ()

  Allocate a block from the pool (without element construction).

- void Deallocate (void *element)

  Deallocate a block and return to the pool (without element destruction).

- size_t GetElemSize () const

  Gets the pool granularity.

- size_t GetSize () const

  Gets the pool size in elements.

- size_t GetUsed () const

  Gets the number of elements allocated from pooled storage.

- T * Construct ()

  Constructs an object from the pool.

- void Destroy (T *instance)

  Destructs an object back into the pool.

9.634.2 Constructor & Destructor Documentation

9.634.2.1 template<typename T > Pool< T >::Pool (size_t *block_size*) [inline]

Allocates a pool with `block_size` elements of `elemsize` bytes each. The pool will grow in increments of `block_size`. Memory is NOT returned to the OS.

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

- het_lib/common/include/pool.h

9.635 camra_handlers::populate_onewire_device_strings Class Reference

Public Member Functions

- **populate_onewire_device_strings** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.635.1 Member Function Documentation

9.635.1.1 void camra_handlers::populate_onewire_device_strings::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.636 populate_onewire_device_strings Class Reference

9.636.1 Detailed Description

Reads all onewire devices in the array, populating related header fields with the information parsed.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.637 pos_fit Class Reference

Public Types

- typedef double **value_type**

Public Member Functions

- double **operator()** (const FVector< double, 2 > &enc_val, const FVector< double, 7 > &p, FVector< double, 7 > &jac) const

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

- het_lib/coord_trans/mount_model/encoder_fit.cpp

9.638 fixed_point::fixed_internal::power10< N > Struct Template Reference

template<unsigned int N> struct fixed_point::fixed_internal::power10< N >

Public Types

- enum { **value** = power10_aux<N, **u** = =0u>::value }

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

- het_lib/common/include/fixed.h

9.639 fixed_point::fixed_internal::power10< 0 > Struct Template Reference

template<> struct fixed_point::fixed_internal::power10< 0 >

Public Types

- enum { **value** = 1u }

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

- het_lib/common/include/fixed.h

9.640 fixed_point::fixed_internal::power10_aux< N, Even > Struct Template Reference

template<unsigned int N, bool Even> struct fixed_point::fixed_internal::power10_aux< N, Even >

Public Types

- enum { **value** = 10u∗power10<N-1u>::value }

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

- het_lib/common/include/fixed.h

9.641 fixed_point::fixed_internal::power10_aux< N, true > Struct Template Reference

template<unsigned int N> struct fixed_point::fixed_internal::power10_aux< N, true >

Public Types

- enum { **value** = square<power10<N/2u>::value>::value }

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

- het_lib/common/include/fixed.h

9.642 PowerACQCamera Class Reference

9.642.1 Detailed Description

Turn on/off the power to the Acquisition Camera.

Required parameters:

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerACQCamera** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.643 PowerBIBCamera Class Reference

9.643.1 Detailed Description

Turn on/off the power to the BIB Camera.

Required parameters:

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerBIBCamera** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.644 PowerHPF_ACQCamera Class Reference

9.644.1 Detailed Description

Turn on/off the power to the HR-SAS Camera.

Required parameters:

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerHPF_ACQCamera** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.645 PowerHR_SASCamera Class Reference

9.645.1 Detailed Description

Turn on/off the power to the HR-SAS Camera.

Required parameters:

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerHR_SASCamera** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.646 PowerOffAll Class Reference

9.646.1 Detailed Description

Turn off the power to the PFIP hardware items: Switch1, Switch2, SBC1, SBC2, Guide Probes, Acquisition Camera, Pupil View/Calibration Wave Front Sensor.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerOffAll** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.647 PowerOnAll Class Reference

9.647.1 Detailed Description

Turn on the power to the PFIP hardware items: Switch1, Switch2, SBC1, SBC2, Guide Probes, Acquisition Camera, Pupil View/CWFS.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerOnAll** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.648 PowerProbeCamera Class Reference

9.648.1 Detailed Description

Turn on/off the power to one of the probes

Required parameters:

camera<string> - One of \"Guider1\", \"Guider2\", \"WFS1\", or \"WFS2\"

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerProbeCamera** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.649 PowerPV_CWFSCamera Class Reference

9.649.1 Detailed Description

Turn on/off the power to the PV and CWFS.

Required parameters:

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerPV_CWFSCamera** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.650 PowerSBC1 Class Reference

9.650.1 Detailed Description

Turn on/off the power to the Single Board Computer 1

Required parameters:

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerSBC1** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.651 PowerSBC2 Class Reference

9.651.1 Detailed Description

Turn on/off the power to the Single Board Computer 2

Required parameters:

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerSBC2** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.652 PowerSwitch1 Class Reference

9.652.1 Detailed Description

Turn on/off the power to the ENet Switch 1

Required parameters:

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerSwitch1** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.653 PowerSwitch2 Class Reference

9.653.1 Detailed Description

Turn on/off the power to the ENet Switch 2

Required parameters:

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerSwitch2** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.654 PowerTTCamera Class Reference

9.654.1 Detailed Description

Turn on/off the power to the HR-SAS Camera.

Required parameters:

state<bool> - true - turn on; false - turn off

`#include <pfip_handler_doc.h>`

Public Member Functions

- **PowerTTCamera** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.655 Probe Class Reference

Public Member Functions

- Probe (const string &sourceName, const string &name, pfip::ProbeID probeID, bool enabled=1)

  Identifies the different probe types.

- virtual ~Probe ()

  Destructor.

- bool CheckNewCurrentEncoderPositions (int &newCarriageEncoder, int &newArmEncoder, ProbePositionCheckAction action)

  Checks newly-read values for sanity, against the known position range for this probe. If they're out of range, replaces values with current positions or start/end values depending on useOld.

- double GetBeginRange ()

  Return the angle of the beginning of this probe's range in the 360 degree assembly in radians.

- double GetEndRange ()

  Return the angle of the end of this probe's range in the 360 degree assembly in radians.

- int GetBeginEncoder ()

  Return the beginning of the encoder range for the probe type.

- int GetEndEncoder ()

  Return the end of the encoder range for the probe type.

- bool ComputeMove (const double x_sphereRadians, const double y_sphereRadians)
- bool **ComputeMove** (const int carriageEncoderCount, const int armEncoderCount)
- ProbePosition * GetCurrentPosition ()

  Return ptr to the ProbePosition object representing the probe's current position.

- ProbePosition * GetTargetPosition ()

  Return ptr to the ProbePosition object representing the probe's target position after move.

- int GetProbeEnd ()

  Return the angle on the 360 degree assembly representing the end of the probe's range.

- int GetProbeStart ()

  Return the encoder range start position for the probe body.

- int GetArmStart ()

  Return the encoder range start position for the probe arm.

- int GetArmEnd ()

  Return the encoder range end position for the probe arm.

- pfip::ProbeID GetID ()

  Return the probe's type attribute.

- void SaveTargetAngles (const double angle, const double actual)

  Set the angles (relative and absolute) for the probe's target position.

- void SaveCurrentAngles (const double angle, const double actual)

  Set the angles(relative and absolute) for the probe's source location position.

- void SetDriveStatus (MAXONDriveStatus &status)

  Store the probe's hardware status bits in the probe object.

- void GetDriveStatus (MAXONDriveStatus &status)

  Return the probe's hardware status bits.

- std::string & GetName ()

  Return the string containing the probe name.

- pfip_coord * GetCoordObj ()

  Return pointer to the coordinate object for this probe.

- void **SetMoving** (bool moving)

Static Public Member Functions

- static std::string GetProbeTypeStr (pfip::ProbeID probe)

  Return a string representing the probe.

Friends

- bool ProbePosition::EncoderCountsToSphereRadians (Probe *probe, const int carriageEncoderCount, const int armEncoderCount)

  Store a new probe state in the probe object.

- bool ProbePosition::SphereRadiansToEncoderCounts (Probe *probe, const double x_sphereRadians, const double y_sphereRadians)

  See ProbePosition class.

- void **ProbePosition::SetEncoderCounts** (const int carriageEncoderCount, const int armEncoderCount)

9.655.1 Constructor & Destructor Documentation

9.655.1.1 Probe::Probe (const string & *sourceName*, const string & *name*, pfip::ProbeID *probeID*, bool *enabled* = 1)

Constructor

9.655.2 Member Function Documentation

9.655.2.1 bool Probe::CheckNewCurrentEncoderPositions (int & *newCarriageEncoder*, int & *newArmEncoder*, ProbePositionCheckAction *action*)

If the positions are outside the valid range by a very small amount, the position will be quietly corrected to the range start or end.

Parameters:

newCarriageEncoder[in,out] The value to be checked for a valid carriage encoder position.

newArmEncoder[in,out] The value to be checked for a valid arm encoder position.

action[in] If ENCPOS_NO_UPDATE, do not change values. If ENCPOS_ADJUST_TO_RANGE, any invalid encoder value will be replaced with the range start or end. If ENCPOS_USE_-CURRENT, invalid values will be replaced with the current position values.

Returns:

True if encoder values were in the valid range for this probe when the call was made. False if either were out of range.

9.655.2.2 bool Probe::ComputeMove (const double *x_sphereRadians*, const double *y_sphereRadians*)

Translate input position locations provided in degrees on the focal surface to encoder values

Parameters:

x_coord[in] - the x-coordinate of the position location

y_coord[in] - the y-coordinate of the position location

9.655.3 Friends And Related Function Documentation

9.655.3.1 bool ProbePosition::EncoderCountsToSphereRadians (Probe * *probe*, const int *carriageEncoderCount*, const int *armEncoderCount*) [friend]

See ProbePosition class

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

- pfip/include/Probe.h
- pfip/src/Probe.cpp

9.656 ProbeAssembly Class Reference

A class to represent the probe assembly. Handles the coordinated movement and collision avoidance for the four probes.

`#include <ProbeAssembly.h>`

Public Member Functions

- ProbeAssembly (const string &sourceName)

  Constructor.

- virtual ~ProbeAssembly ()

  Destructor.

- bool Initialize (EPICSComm *epicsComm, PFIPConfig *const config)

  Allocate and initialize the probe objects in the assembly.

- int **MoveProbes** (EPICSComm *epics)
- bool ComputeTargetPositions (const MoveProbeParms(&moveParms)[pfip::NUMBER_OF_PROBES])

  Compute the relative angular positions of the probes, given their target coordinates.

- bool EnableDisableProbes (EPICSComm *const epicsComm, const bool enableFlag)

  Enable or disable all configured probes.

- bool GetProbeInfo (EPICSComm *epicsComm, ProbeInfo **info)

  return current status and holding values for probe source positions (before move)

- bool GetAllProbeEncoderCounts (EPICSComm *epicsComm, int *encoderCounts, bool bSendEvent=0)

  Read EPICS encoder values for all probes and return settings in an array.

- bool UpdateProbePositions (EPICSComm *epicsComm, bool updateMoveEventPayload=0, string position="")

  Initialize probe status and location information in probe assembly object from hardware settings.

- void InterruptRequest ()

  Interrupt a move probes or enable/disable probes operation.

- bool ReadStatusValues (EPICSComm *epicsComm)

  Read status values from EPICS for each probe and store in the probe's status object.

- void SetPower (EPICSComm *epicsComm, const pfip::ProbeID probe, const pfip::PowerSetting setting)

  Turn the probe power on/off.

- pfip::PowerSetting GetProbePower (const pfip::ProbeID probe)

  Read the hardware power settings for a probe.

- bool **ChangeGain** (EPICSComm *epicsComm, const pfip::ProbeID probeID, int IGain, int PGain, bool bLockEPICS=1)
- bool **TimeoutNextMove** (EPICSComm *epicsComm, const double timeout)
- void **InitMoveEvent** (const string &handlerName)
- void **SendMoveEvent** (const string &handlerName)
- void **MarkMoveEventComplete** (int retVal)

9.656.1 Member Function Documentation

9.656.1.1 bool ProbeAssembly::EnableDisableProbes (EPICSComm *const *epicsComm*, const bool *enableFlag*)

Parameters:

epicsComm[in] - ptr to EPICS communication object

enableFlag - false == disable, true == enable Returns 1 if successful, 0 if error occurred, -1 if interrupted

9.656.1.2 bool ProbeAssembly::GetAllProbeEncoderCounts (EPICSComm * *epicsComm*, int * *encoderCounts*, bool *bSendEvent* = 0)

Parameters:

epicsComm[in] - ptr to EPICS communication object

encoderCounts[out] - ptr to array of values to be returned (8 items, x and y positions for each guider)

9.656.1.3 bool ProbeAssembly::GetProbeInfo (EPICSComm * *epicsComm*, ProbeInfo ** *info*)

Parameters:

epicsComm[in] - ptr to EPICS communication object

info[out] - structure to store information returned

9.656.1.4 bool ProbeAssembly::Initialize (EPICSComm * *epicsComm*, PFIPConfig *const *config*)

Parameters:

epicsComm[in] - ptr to the EPICS communication object

9.656.1.5 bool ProbeAssembly::ReadStatusValues (EPICSComm * *epicsComm*)

Parameters:

epicsComm[in] - ptr to the EPICS communication object

9.656.1.6 bool ProbeAssembly::UpdateProbePositions (EPICSComm * *epicsComm*, bool *updateMoveEventPayload* = 0, string *position* = "")

Parameters:

epicsComm[in] - ptr to the EPICS communication object

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

- pfip/include/ProbeAssembly.h
- pfip/src/ProbeAssembly.cpp

9.657 probeAssemblyScene Class Reference

Public Member Functions

- **probeAssemblyScene** (QGraphicsView ∗pView, probeAssemblyTabUi ∗pTab)
- void **setProbeAngle** (pfip::ProbeID probeID, const double angle)
- void **setProbeOrientation** (pfip::ProbeID probeID, const double angle)
- void **setArmAngle** (pfip::ProbeID probeID, const double angle)
- void **setX** (pfip::ProbeID probeID, const double x)
- void **setY** (pfip::ProbeID probeID, const double y)
- void **setColor** (pfip::ProbeID probeID, const QColor &c)
- float **getProbeAngle** (pfip::ProbeID probeID)
- void **drawProbeRanges** (void)

Public Attributes

- bool **m_showRanges** [pfip::NUMBER_OF_PROBES]

Protected Attributes

- QRectF ∗ **m_guiderRect**
- QRectF ∗ **m_innerRect**
- QRectF ∗ **m_outerRect**
- QGraphicsView ∗ **m_pView**
- probeAssemblyTabUi ∗ **m_pTab**
- QList< QGraphicsItem ∗ > **m_probeRangePolys**

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

- gui/include/probeAssembly.h
- gui/src/probeAssembly.cpp

9.658 Ui::probeAssemblyTab Class Reference

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

- gui/build/ui_probeAssemblyTab.h

9.659 probeAssemblyTabUi Class Reference

Public Slots

- void **SubSystemRunning** (std::string, bool)
- void **UpdateProbeStatus** (pfip::ProbeID)
- void **SetDrawProbeRanges** (bool state)

Signals

- void **ForceHeartbeat** (std::string)

Public Member Functions

- **probeAssemblyTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)
- bool **eventFilter** (QObject *pObject, QEvent *pEvent)
- void **resize** (void)

Protected Attributes

- probeAssemblyScene * **m_pProbeAssemblyScene**

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

- gui/include/probeAssemblyTabUi.h
- gui/build/moc_probeAssemblyTabUi.cpp
- gui/src/probeAssemblyTabUi.cpp
- playground/probepointgen/gui/probeAssemblyTabUi.cpp

9.660 Ui::probeControlTab Class Reference

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

- gui/build/ui_probeControlTab.h

9.661 probeControlTabUi Class Reference

Signals

- void **IssueMoveProbes** (QString, QString, QString, QString, QString, QString, QString, QString)
- void **ForceHeartbeat** (std::string)

Public Member Functions

- **probeControlTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

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

- gui/include/probeControlTabUi.h
- gui/build/moc_probeControlTabUi.cpp
- gui/src/probeControlTabUi.cpp

9.662 probeData Struct Reference

Public Member Functions

- void **getResiduals** (const std::list< point > &pl, const char filename[100], const char filename2[100])
- void **enc_to_angles** (const double enc_t, const double enc_p, double &beta, double &gma)
- ltl::FVector< double, 3 > **Rcomp** (const double beta, const double gma)

Public Attributes

- double **armLen**
- double **trackRad**
- ltl::FVector< double, 3 > **centre**
- double **beta_min**
- double **gamma_min**
- double **enc_t_min**
- double **enc_t_max**
- double **enc_p_min**
- double **enc_p_max**
- double **mx_a0**
- double **mx_a1**
- double **mx_a2**
- double **mx_a3**
- double **bx_a0**
- double **bx_a1**
- double **bx_a2**
- double **bx_a3**
- double **my_a0**
- double **my_a1**
- double **my_a2**
- double **my_a3**
- double **by_a0**
- double **by_a1**
- double **by_a2**
- double **by_a3**

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

- het_lib/coord_trans/mount_model/encoder_fit.cpp

9.663 Ui::probeGainControlTab Class Reference

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

- gui/build/ui_probeGainControlTab.h

9.664 probeGainControlTabUi Class Reference

Signals

- void **IssueSetProbeGains** (QString, QString, QString, QString, QString, QString, QString, QString)
- void **ForceHeartbeat** (std::string)

Public Member Functions

- **probeGainControlTabUi** (const char ∗name, QWidget ∗pLoadedWidget, guiConfig ∗pConfig)

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

- gui/include/probeGainControlTabUi.h
- gui/build/moc_probeGainControlTabUi.cpp
- gui/src/probeGainControlTabUi.cpp

9.665 ProbeInfo Class Reference

Public Member Functions

- **ProbeInfo** (pfip::ProbeID p)
- void **SendProbeStatusEvent** ()
- Message * **ReplyProbePosition** ()
- Message * **ReplyProbeInfo** ()
- void **SetDriveStatus** (MAXONDriveStatus &s)
- void **SetFilterStatus** (MAXONDriveStatus &s)
- void **SetDriveData** (MAXONDriveData &d)
- void **SetAngleValues** (Probe *p, int probEncoder, int armEncoder)
- void **SetPower** (bool p)
- void **SetFilterPosition** (int i)
- void **SetFilterName** (std::string name)
- void **SetFilterConfig** (std::string configs)

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

- pfip/include/PFIPStatus.h
- pfip/src/EventInfo.cpp

9.666 ProbeItem Class Reference

Public Member Functions

- **ProbeItem** (QGraphicsEllipseItem ∗const item, QGraphicsSimpleTextItem ∗const label, QGraphicsLineItem ∗const arm, QGraphicsLineItem ∗const arrowLine, const float armLen, const float angle, const float m_armangle, const float radius, QGraphicsScene ∗scene)
- void **setProbeAngle** (const float angle_degrees, QGraphicsEllipseItem ∗const assembly, const float assemblyRadius)
- void **setArmAngle** (const float angle_degrees)
- void **setProbeOrientation** (const float angle_degrees, QGraphicsEllipseItem ∗const assembly, const float assemblyRadius)
- void **setProbeColor** (const QColor &c)
- float **getProbeAngle** (void)
- void **setProbeX** (const float x)
- void **setProbeY** (const float y)
- **ProbeItem** (QGraphicsEllipseItem ∗const item, QGraphicsSimpleTextItem ∗const label, QGraphicsLineItem ∗const arm, const float armLen, const float angle, const float arm_angle, const float radius, QGraphicsScene ∗scene)
- void **setGuiderAngle** (const float angle, QGraphicsEllipseItem ∗const assembly, const float assemblyRadius)
- void **setGuiderArmAngle** (const float armAngle)
- void **setGuiderColor** (const QColor &c)
- float **getGuiderAngle** ()
- float **getGuiderArmAngle** ()
- void **setGuiderX** (const float X)
- void **setGuiderY** (const float Y)
- float **getGuiderX** ()
- float **getGuiderY** ()

Public Attributes

- float **m_probeAngle_radians**
- float **m_armAngle_radians**
- QColor **m_probeColor**
- double **m_arrowAngle**
- double **m_cosArrowAngle**
- double **m_sinArrowAngle**
- float **guiderAngle**
- float **armAngle**
- QColor **guiderColor**

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

- gui/include/probeAssembly.h
- playground/probepointgen/gui/qTcsProbeAssembly.h
- gui/src/probeAssembly.cpp
- playground/probepointgen/gui/qTcsProbeAssembly.cpp

9.667 ProbePosition Class Reference

Public Member Functions

- ProbePosition (const string &sourceName)

  Constructor.

- virtual ~ProbePosition ()

  Destructor.

- double GetProbeActualAngleRadians ()

  Return the angle of the probe position within the circular assembly.

- double GetAngleRadians (WhichMotor which)

  Return angle of the probe arm.

- int GetEncoderCount (WhichMotor which)

  Initialize the object with.

- void **SetEncoderCount** (WhichMotor which, const int pos)
- bool SphereRadiansToEncoderCounts (Probe *p, const double x_sphereRadians, const double y_sphereRadians)

  Return the position's y-coordinate.

- bool EncoderCountsToSphereRadians (Probe *p, const int probe_encoderCount, const int arm_encoderCount)

  Translate encoder values to angles of the probe position.

- void **SetEncoderCounts** (const int prob_encodeCount, const int arm_encoderCount)
- double GetTelecenAngleRadians (WhichMotor which)

  Set the position's start variable.

Protected Attributes

- int **encoderCount_** [2]
- double probeActualAngleRadians_

  Relative position of guider in assembly.

9.667.1 Member Function Documentation

9.667.1.1 int ProbePosition::GetEncoderCount (WhichMotor *which*) `[inline]`

Parameters:

x_coord[in] - x coordinate of the position

y_coord[in] - y coordinate of the position

beginRange[in] - beginning of the range within the circular assembly

endRange[in] - end of the range within the circular assembly

encoderBegin[in] - the encoder value for the start of the range

encoderEnd[in] - the encoder value for the end of the range Return the position's x-coordinateq

9.667.1.2 bool ProbePosition::SphereRadiansToEncoderCounts (Probe * *p*, const double *x_sphereRadians*, const double *y_sphereRadians*)

Store the guider angles in the position object

Parameters:

angle[in] - the angle of the position within the 180 degree range

actual[in] - the angle of the position within the 360 degree range of assembly Translate angles of probe position to raw encoder values

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

- pfip/include/Probe.h
- pfip/src/Probe.cpp

9.668 ProbeStatus Class Reference

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- status **m_Probe**

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

- gui/include/pfipStatusData.h
- gui/src/pfipStatusData.cpp

9.669 Ui::probeStatusTab Class Reference

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

- gui/build/ui_probeStatusTab.h

9.670 probeStatusTabUi Class Reference

Public Slots

- void **SubSystemRunning** (std::string, bool)
- void **UpdateProbeStatus** (pfip::ProbeID)

Signals

- void **ForceHeartbeat** (std::string)

Public Member Functions

- **probeStatusTabUi** (const char ∗name, QWidget ∗pLoadedWidget, guiConfig ∗pConfig)

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

- gui/include/probeStatusTabUi.h
- gui/build/moc_probeStatusTabUi.cpp
- gui/src/probeStatusTabUi.cpp

9.671 ProcessorConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- pas::ProcessingMode_t **m_processingMode**
- string **m_pupilgeom**
- string **m_gridgeom**
- string **m_imagemask**
- string **m_modalcube**
- string **m_trkergeom**
- string **m_wfcm4geom**
- string **m_spidrgeom**
- string **m_m1geom**
- string **m_calibdata**
- float **m_pix2mm**
- float **m_pix2rad**
- float **m_wavelen**
- float **m_wfsclock**
- float **m_TT2xyoffset**
- float **m_DF2woffset**
- float **m_CM2tpoffset**
- int **m_xSkyOffsetSign**
- int **m_ySkyOffsetSign**
- int **m_wFocusOffsetSign**
- int **m_thetaOffsetSign**
- int **m_phiOffsetSign**
- float **m_M1xmotion2itf_theta**
- float **m_M1xmotion2itf_phi**
- float **m_M1ymotion2itf_theta**
- float **m_M1ymotion2itf_phi**
- float **m_TRKxmotion2itf_theta**
- float **m_TRKxmotion2itf_phi**
- float **m_TRKymotion2itf_theta**
- float **m_TRKymotion2itf_phi**
- float **m_ROTmotion2itf_rho**
- float **m_ROTmotion2prb_crg**
- float **m_ROTmotion2prb_arm**
- float **m_M4motion2prb_telx**
- float **m_M4motion2prb_tely**
- float **m_xSkyOffsetCalibration**
- float **m_ySkyOffsetCalibration**
- float **m_wFocusOffsetCalibration**
- float **m_thetaOffsetCalibration**
- float **m_phiOffsetCalibration**

- float **m_probeCarriageAngleZeropoint**
- float **m_probeArmAngleZeropoint**
- int **m_cenmethod**
- int **m_threshmethod**
- int **m_niter**
- int **m_PSFmodel**
- float **m_PSFdia**
- float **m_thresh_percent**
- float **m_thresh_nsigma**
- float **m_wfs_gain**
- float **m_wfs_flen**
- float **m_magZero**
- float **m_detectSigma**
- int **m_minPixels**
- int **m_maxPixels**
- float **m_gain**
- float **m_nanValue**
- float **m_saturation**
- int **m_skyOrder**
- int **m_skyAnnulus**
- float **m_kronAperture**
- float **m_fixedAperture**

9.671.1 Member Function Documentation

9.671.1.1 void ProcessorConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

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

- pas/include/pasConfig.h
- pas/src/pasConfig.cpp

9.672 ProcessVariable< T > Class Template Reference

template<typename T> class ProcessVariable< T >

Public Member Functions

- **ProcessVariable** (const std::string &name)
- **ProcessVariable** (const std::string &name, const std::string &subfields)
- virtual void **apply_parms** (Payload &p) const
- virtual void **set** (const void *raw)
- virtual void **set** (const unsigned n, const void *raw)
- void **update_time** (void)
- template<>
  void **set** (const void *raw)
- template<>
  void **set** (const void *raw)
- template<>
  void **set** (const void *raw)
- template<>
  void **set** (const void *raw)
- template<>
  void **set** (const void *raw)
- template<>
  void **set** (unsigned n, const void *raw)
- template<>
  void **set** (unsigned n, const void *raw)
- template<>
  void **set** (unsigned n, const void *raw)
- template<>
  void **set** (unsigned n, const void *raw)
- template<>
  void **set** (unsigned n, const void *raw)
- template<>
  ProcessVariable (const std::string &name, const std::string &subfields)
- template<>
  ProcessVariable (const std::string &name, const std::string &subfields)
- template<>
  ProcessVariable (const std::string &name, const std::string &subfields)
- template<>
  ProcessVariable (const std::string &name, const std::string &subfields)
- template<>
  void **apply_parms** (Payload &p) const
- template<>
  void **apply_parms** (Payload &p) const
- template<>
  void **apply_parms** (Payload &p) const
- template<>
  void **apply_parms** (Payload &p) const
- template<>
  void **apply_parms** (Payload &p) const
- template<>
  void **apply_parms** (Payload &p) const

- template<>
  void **apply_parms** (Payload &p) const

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

- epics_relay/process_variable.h

9.673 Ui::productionTab Class Reference

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

- gui/build/ui_productionTab.h

9.674 productionTabUi Class Reference

Public Slots

- void **UpdateApcStatus** (std::string, std::string, int, int, double, int, int)
- void **UpdateTrackerPosition** (void)

Signals

- void **ForceHeartbeat** (std::string)
- void **ForceLegacyStatusEvents** (void)
- void **ForcePipelineStatusEvent** (std::string)
- void **MoveDomeTo** (double)
- void **MoveStructureToTrueAz** (double)
- void **SetDomeDirTo** (std::string)
- void **EnableSACDMIMeasurements** (std::string, bool)
- void **EnableCCASDMIMeasurements** (std::string, bool)
- void **MoveTrackerITF** (std::string, std::string, double, double, double, double, double, double)
- void **MoveTrackerRTF** (std::string, std::string, double, double, double, double, double, double)
- void **TurnOnOutletByName** (std::string)
- void **TurnOffOutletByName** (std::string)
- void **LoadRaDecTrajectory** (double, double, double, std::string, double, double)
- void **LoadXYTrajectory** (double, double, double)
- void **LoadFileTrajectory** (std::string, double, double)
- void **LoadSatelliteTrajectory** (std::string, std::string, double)
- void **LoadEphemerisTrajectory** (std::string, double, std::string)
- void **GoNext** (bool, bool, bool, bool)
- void **OffsetTrajectoryTipTilt** (double, double)
- void **OffsetTrajectoryRaDec** (double, double, bool)
- void **OffsetTrajectoryXY** (double, double, bool)
- void **OffsetTrajectoryRho** (double)
- void **OffsetFocus** (double)
- void **SetStoreImages** (std::string, bool)
- void **RelayCameraMetrologyControlCommand** (std::string, std::string)
- void **StartPipeline** (std::string)
- void **StopPipeline** (std::string)
- void **IssuePFIPControl** (QString, QString, bool)
- void **SendPFIPCommand** (const std::string, Message ∗)
- void **SetACQCameraFilter** (std::string)
- void **DeployACQCameraMirror** (void)
- void **RetractACQCameraMirror** (void)
- void **DeployPV_CWFSCameraMirror** (void)
- void **RetractPV_CWFSCameraMirror** (void)
- void **DeployPVCamera** (void)
- void **DeployCWFSCamera** (void)

Public Member Functions

- **productionTabUi** (const char ∗name, QWidget ∗pLoadedWidget, guiConfig ∗pConfig)
- void **loadSettings** (void)
- void **saveSettings** (void)

Protected Member Functions

- bool **constructApcTab** (void)
- void **LoadRaDecTrajectory** (void)
- void **LoadXYTrajectory** (void)
- void **LoadFileTrajectory** (void)
- void **LoadSatelliteTrajectory** (void)
- void **LoadEphemerisTrajectory** (void)
- bool **BrowseForFile** (QString &file)

Protected Attributes

- QString **m_filterNamesACQ**
- QString **m_filterNamesG1**
- QString **m_filterNamesG2**

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

- gui/include/productionTabUi.h
- gui/build/moc_productionTabUi.cpp
- gui/src/productionTabUi.cpp

9.675 proxy Class Reference

9.675.1 Detailed Description

Executes the configured command on the given args.

Optional parameters:

args<string> - Argument list string that will be passed to the configured command.

`#include <tcs_proxy_handler_doc.h>`

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

- proxy/tcs_proxy_handler_doc.h

9.676 Proxy Class Reference

Public Member Functions

- **Proxy** (const string &appName, ProxyConfig ∗c)
- virtual int **start** ()
- virtual void **stop** ()
- std::string **exec** (std::string &command, int &rc)
- void **respond** (Message ∗message, std::string &response)
- void **log** (std::string &response)
- void **execute_callback** (const std::string &args, Message ∗response=NULL)

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

- proxy/tcs_proxy.h
- proxy/tcs_proxy.cpp

9.677 ProxyConfig Class Reference

Public Member Functions

- **ProxyConfig** (bool is_named=0)
- virtual void print_summary (void)

  Called prior to displaying options.

- virtual void set_options (OptionParser &p)

  Adds options to default set.

Public Attributes

- std::string **proxy_route_**
- std::string **proxy_event_route_**
- std::string **proxy_command_**
- std::string **proxy_source_**
- std::string **proxy_key_**
- bool **emit_events_**
- bool **log_results_**
- float **rate_hertz_**

9.677.1 Member Function Documentation

9.677.1.1 void ProxyConfig::print_summary (void) [virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.677.1.2 void ProxyConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- proxy/tcs_proxy.h
- proxy/tcs_proxy.cpp

9.678 ProxyException Class Reference

Public Member Functions

- **ProxyException** (tcsexcept::tcs_exception_t __proxy, const std::string &w, const char ∗file, int line, const char ∗func)

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

- het_lib/common/include/tcs_exception.h

9.679 ProxyHandler Class Reference

Public Member Functions

- **ProxyHandler** (Proxy &proxy, const std::string &command)
- virtual void process (Message *m)

9.679.1 Member Function Documentation

9.679.1.1 void ProxyHandler::process (Message * *message*) [virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- proxy/tcs_proxy.h
- proxy/tcs_proxy.cpp

9.680 PSFFitData Class Reference

Public Member Functions

- virtual void **newData** (int size)
- virtual void **deleteData** (void)

Public Attributes

- size_t **n**
- double ∗ **r**
- double ∗ **y**
- size_t **psfkind**

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

- het_lib/wfs_lib/include/wfslib.h

9.681 PupilData Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual void **newData** (int size)
- virtual void **deleteData** (void)

Public Attributes

- ushort **Nvertx**
- float **ri**
- float **ro**
- float **e**
- float **x0**
- float **y0**
- float * **Xouter_vertex**
- float * **Youter_vertex**
- float * **Xinner_vertex**
- float * **Yinner_vertex**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.682 QBlinkingLineEdit Class Reference

Public Member Functions

- **QBlinkingLineEdit** (QWidget ∗pParent=0)
- QSize **minimumSizeHint** () const
- QSize **sizeHint** () const
- int **blinkRate** (void)
- void **setBlinkRate** (int rate)
- QColor **normalFGColor** (void)
- void **setNormalFGColor** (QColor color)
- QColor **normalBGColor** (void)
- void **setNormalBGColor** (QColor color)
- QColor **reverseFGColor** (void)
- void **setReverseFGColor** (QColor color)
- QColor **reverseBGColor** (void)
- void **setReverseBGColor** (QColor color)
- void **setColors** (QColor normBG, QColor normFG, QColor revBG, QColor revFG)
- void **setBlinking** (bool blink)

Properties

- int **blinkRate**
- QColor **normalFGColor**
- QColor **normalBGColor**
- QColor **reverseFGColor**
- QColor **reverseBGColor**

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

- gui/designer/customWidgetPlugins/blinkingLineEdit/qBlinkingLineEdit.h
- gui/designer/customWidgetPlugins/blinkingLineEdit/qBlinkingLineEdit.cpp

9.683 QBlinkingLineEditPlugin Class Reference

Public Member Functions

- **QBlinkingLineEditPlugin** (QObject ∗parent=NULL)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **domXml** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget ∗ **createWidget** (QWidget ∗parent)
- void **initialize** (QDesignerFormEditorInterface ∗core)

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

- gui/designer/customWidgetPlugins/blinkingLineEdit/qBlinkingLineEditPlugin.h
- gui/designer/customWidgetPlugins/blinkingLineEdit/qBlinkingLineEditPlugin.cpp

9.684 QGraphicsCircleItem Class Reference

Public Member Functions

- **QGraphicsCircleItem** (const QPointF center, const qreal radius, QGraphicsItem ∗parent=0, QGraphicsScene ∗scene=0)

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

- gui/include/focalPlaneShutter.h
- gui/src/focalPlaneShutter.cpp

9.685 QLedIndicator Class Reference

Public Types

- enum **ledState** { **ledOn**, **ledOff**, **ledUndef** }
- enum **labelPosition** { **labelLeft**, **labelTop**, **labelRight**, **labelBottom** }

Signals

- void **clicked** (bool checked=false)

Public Member Functions

- **QLedIndicator** (QWidget ∗parent)
- void **setDiameter** (int diameter)
- void **setState** (ledState state)
- void **setLabel** (QString label)
- void **setLabelPosition** (labelPosition position)
- void **setText** (QString text)
- void **setForegroundColor** (QColor color)
- void **setBackgroundColor** (QColor color)
- void **setOnColor** (QColor color)
- void **setOffColor** (QColor color)
- void **setUndefColor** (QColor color)
- void **setFlashColor** (QColor color)
- void **setFlashRate** (int rate)
- void **setFlashing** (bool flashing)
- int **getDiameter** (void)
- ledState **getState** (void)
- QString **getLabel** (void)
- labelPosition **getLabelPosition** (void)
- QString **getText** (void)
- QColor **getForegroundColor** (void)
- QColor **getBackgroundColor** (void)
- QColor **getOnColor** (void)
- QColor **getOffColor** (void)
- QColor **getUndefColor** (void)
- QColor **getFlashColor** (void)
- int **getFlashRate** (void)
- bool **getFlashing** (void)

Protected Member Functions

- virtual void **paintEvent** (QPaintEvent ∗event)
- virtual void **mouseReleaseEvent** (QMouseEvent ∗event)

Properties

- int **diameter**
- ledState **state**
- QString **label**
- QString **text**
- labelPosition **labelPosition**
- QColor **foregroundColor**
- QColor **backgroundColor**
- QColor **onColor**
- QColor **offColor**
- QColor **undefColor**
- QColor **flashColor**
- int **flashRate**
- bool **flashing**

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

- gui/designer/customWidgetPlugins/ledIndicator/qLedIndicator.h
- gui/designer/customWidgetPlugins/ledIndicator/build/moc_qLedIndicator.cpp
- gui/designer/customWidgetPlugins/ledIndicator/qLedIndicator.cpp

9.686 QLedIndicatorPlugin Class Reference

Public Member Functions

- **QLedIndicatorPlugin** (QObject ∗parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **domXml** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget ∗ **createWidget** (QWidget ∗parent)
- void **initialize** (QDesignerFormEditorInterface ∗core)

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

- gui/designer/customWidgetPlugins/ledIndicator/qLedIndicatorPlugin.h
- gui/designer/customWidgetPlugins/ledIndicator/qLedIndicatorPlugin.cpp

9.687 QSparkLine Class Reference

Public Member Functions

- **QSparkLine** (QWidget ∗parent)
- void **setLineColor** (QColor color)
- void **setPointColor** (QColor color)
- void **setPlotLineWidth** (int width)
- void **setMin** (qreal min)
- void **setMax** (qreal max)
- QColor **getLineColor** (void)
- QColor **getPointColor** (void)
- int **getPlotLineWidth** (void)
- qreal **getMin** (void)
- qreal **getMax** (void)
- void **addNewValue** (const qreal value)
- void **addNewValueAutoScale** (const qreal value)

Protected Member Functions

- virtual void **paintEvent** (QPaintEvent ∗pEvent)
- void **addValue** (const qreal value)

Properties

- QColor **lineColor**
- QColor **pointColor**
- int **plotLineWidth**
- qreal **min**
- qreal **max**

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

- gui/designer/customWidgetPlugins/sparkLine/qSparkLine.h
- gui/designer/customWidgetPlugins/sparkLine/qSparkLine.cpp

9.688 QSparkLinePlugin Class Reference

Public Member Functions

- **QSparkLinePlugin** (QObject ∗parent=NULL)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **domXml** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget ∗ **createWidget** (QWidget ∗parent)
- void **initialize** (QDesignerFormEditorInterface ∗core)

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

- gui/designer/customWidgetPlugins/sparkLine/qSparkLinePlugin.h
- gui/designer/customWidgetPlugins/sparkLine/qSparkLinePlugin.cpp

9.689 QTcsFrame Class Reference

Public Member Functions

- **QTcsFrame** (QWidget ∗pParent=0)
- void **setTitle** (QString title)
- void **setTitleBackgroundColor** (QColor color)
- void **setTitleColor** (QColor color)
- QString **getTitle** (void)
- QColor **getTitleBackgroundColor** (void)
- QColor **getTitleColor** (void)

Protected Member Functions

- virtual void **paintEvent** (QPaintEvent ∗event)

Protected Attributes

- QString **m_title**
- QColor **m_titleBackgroundColor**
- QColor **m_titleColor**

Properties

- QString **title**
- QColor **titleBackgroundColor**
- QColor **titleColor**

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

- gui/designer/customWidgetPlugins/tcsFrame/qTcsFrame.h
- gui/designer/customWidgetPlugins/tcsFrame/qTcsFrame.cpp

9.690 QTcsFramePlugin Class Reference

Public Member Functions

- **QTcsFramePlugin** (QObject ∗parent=NULL)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon **icon** () const
- QString **domXml** () const
- QString **group** () const
- QString **includeFile** () const
- QString **name** () const
- QString **toolTip** () const
- QString **whatsThis** () const
- QWidget ∗ **createWidget** (QWidget ∗parent)
- void **initialize** (QDesignerFormEditorInterface ∗core)

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

- gui/designer/customWidgetPlugins/tcsFrame/qTcsFramePlugin.h
- gui/designer/customWidgetPlugins/tcsFrame/qTcsFramePlugin.cpp

9.691 QTcsProbeAssembly Class Reference

Public Member Functions

- **QTcsProbeAssembly** (QWidget *pParent=0)
- void **setTitle** (QString title)
- void **setTitleBackgroundColor** (QColor color)
- void **setTitleColor** (QColor color)
- void **setguider1Angle** (const double guiderAngle)
- void **setguider1X** (const double X)
- void **setguider1Y** (const double Y)
- void **setguider2X** (const double X)
- void **setguider2Y** (const double Y)
- void **setwfs1X** (const double X)
- void **setwfs1Y** (const double Y)
- void **setwfs2X** (const double X)
- void **setwfs2Y** (const double Y)
- void **setguider1ArmAngle** (const double armAngle)
- void **setguider2Angle** (const double guiderAngle)
- void **setguider2ArmAngle** (const double armAngle)
- void **setwfs1Angle** (const double guiderAngle)
- void **setwfs1ArmAngle** (const double armAngle)
- void **setwfs2Angle** (const double guiderAngle)
- void **setwfs2ArmAngle** (const double armAngle)
- void **setguider1Color** (const QColor &c)
- void **setguider2Color** (const QColor &c)
- void **setwfs1Color** (const QColor &c)
- void **setwfs2Color** (const QColor &c)
- QString **getTitle** (void)
- QColor **getTitleBackgroundColor** (void)
- QColor **getTitleColor** (void)
- float **getguider1Angle** (void)
- float **getwfs1Angle** (void)
- float **getguider2Angle** (void)
- float **getwfs2Angle** (void)
- float **getguider1ArmAngle** (void)
- float **getwfs1ArmAngle** (void)
- float **getguider2ArmAngle** (void)
- float **getwfs2ArmAngle** (void)
- float **getguider1X** (void)
- float **getguider1Y** (void)
- float **getguider2X** (void)
- float **getguider2Y** (void)
- float **getwfs1X** (void)
- float **getwfs1Y** (void)
- float **getwfs2X** (void)
- float **getwfs2Y** (void)
- QColor **getguider1Color** (void)
- QColor **getguider2Color** (void)
- QColor **getwfs1Color** (void)
- QColor **getwfs2Color** (void)

Protected Attributes

- QString **m_title**
- QColor **m_titleBackgroundColor**
- QColor **m_titleColor**
- QRectF * **m_guiderRect**
- QRectF * **m_innerRect**
- QRectF * **m_outerRect**
- QGraphicsScene **scene**

Properties

- QString **title**
- QColor **titleBackgroundColor**
- QColor **titleColor**
- double **guider2Angle**
- double **guider2TelecenX**
- double **guider2TelecenY**
- QColor **guider2Color**
- double **wfs1Angle**
- double **wfs2Angle**
- double **guider1TelecenX**
- double **guider1TelecenY**
- double **wfs1TelecenX**
- double **wfs1TelecenY**
- double **wfs2TelecenX**
- double **wfs2TelecenY**
- QColor **guider1Color**
- QColor **wfs1Color**
- double **guider1Angle**
- QColor **wfs2Color**

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

- playground/probepointgen/gui/qTcsProbeAssembly.h
- playground/probepointgen/gui/qTcsProbeAssembly.cpp

9.692 quadFit Class Reference

Public Types

- typedef double **value_type**

Public Member Functions

- double **operator()** (const double &x, const ltl::FVector< double, 3 > &p, ltl::FVector< double, 3 > &jac) const

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

- het_lib/coord_trans/mount_model/curveFits.h

9.693 Quaternion< real > Class Template Reference

template<typename real> class Quaternion< real >

Public Member Functions

- **Quaternion** (real s, real x, real y, real z)
- **Quaternion** (real s)
- **Quaternion** (real angle, real unitAxis[3])
- void **Set** (real s, real x, real y, real z)
- real **Gets** ()
- real **Getx** ()
- real **Gety** ()
- real **Getz** ()
- Quaternion **operator+** (Quaternion q2) const
- Quaternion **operator-** (Quaternion q2) const
- Quaternion **operator*** (Quaternion q2) const
- Quaternion **operator/** (Quaternion q2) const
- Quaternion **conj** ()
- Quaternion & **operator=** (Quaternion rhs)
- Quaternion & **operator=** (real s_)
- int **operator==** (Quaternion rhs)
- int **operator!=** (Quaternion rhs)
- void **Normalize** ()
- void **MoveToRightHalfSphere** ()
- real **Norm2** ()
- real **Norm** ()
- void **Quaternion2Matrix** (real *R)
- void **GetRotation** (real *angle, real unitAxis[3])
- void **GetSinExponential** (real *x, real *y, real *z)
- void **Print** ()

Static Public Member Functions

- static Quaternion **Matrix2Quaternion** (real *R)

Protected Attributes

- real **s**
- real **x**
- real **y**
- real **z**

Friends

- Quaternion< real > **operator*** (real alpha, Quaternion< real > q2)

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

- het_lib/common/include/quaternion.h

9.694 query_trajectory Class Reference

9.694.1 Detailed Description

Calculates a trajectory based on the given parameters and returns the trajectory descriptor. Overloads are available for sidereal and non-sidereal trajectories. Parameters are identical to those of load_trajectory.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **query_trajectory** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.694.2 Member Function Documentation

9.694.2.1 void query_trajectory::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.695 tthread::ratio< N, D > Class Template Reference

9.695.1 Detailed Description

template<__intmax_t N, __intmax_t D = 1> class tthread::ratio< N, D >

Minimal implementation of the `ratio` class. This class provides enough functionality to implement some basic `chrono` classes.

`#include <tinythread.h>`

Static Public Member Functions

- static double **_as_double** ()

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

- het_lib/tthread/include/tinythread.h

9.696 camra_handlers::read_now Class Reference

Public Member Functions

- **read_now** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.696.1 Member Function Documentation

9.696.1.1 void camra_handlers::read_now::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.697 read_now Class Reference

9.697.1 Detailed Description

Ends the currenlty active exposure and initiate readout.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.698 ReadRegister Class Reference

9.698.1 Detailed Description

Read the contents of a modbus register for the pfip enclosure PLC and print the value in the pfip log.

Required parameters:

regnum<int> - The modbus register number to read. These numbers can be found in the PFIP enclosure hardware documentation.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **ReadRegister** (Receiver ∗rcv, ACQ_SubSystem ∗acq)
- virtual void HandlerAction (Message ∗pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message ∗pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.699 Receiver Class Reference

9.699.1 Detailed Description

Receiver class providing an object invocation service for one or more Sender instances.

`#include <receiver.h>`

Public Member Functions

- Receiver (const std::string &endpoint, const std::string &name)
- bool start (void)
- void stop (void)
- bool registerHandler (const std::string &name, MessageHandler *handler, std::string usage="", bool reentrant=false)
- void **registerHandler** (MessageHandler *handler)

Friends

- class Message
- class InterruptHandler
- class HelpHandler
- class StatusHandler
- class UptimeHandler
- void **receiver_recv_thread** (void *z, zctx_t *ctx, void *pipe)
- int **receiver_shutdown_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **receiver_recv_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **receiver_run_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)

9.699.2 Constructor & Destructor Documentation

9.699.2.1 Receiver::Receiver (const std::string & *endpoint*, const std::string & *name*)

Receiver constructor taking a `endpoint` on which to listen for incoming Sender connections and a `name` that will be used to identify outgoing message traffic.

The endpoint is a string of the form "<transport>://<address>" Transport can be one of "tcp", "ipc", "inproc", or "pgm". The address is transport-specific. In most cases with HET process communication, the endpoint of of type tcp and is specified as: "tcp://127.0.0.1:8888" where the 127.0.0.1:8888 specifies an ip address and a port.

9.699.3 Member Function Documentation

9.699.3.1 bool Receiver::registerHandler (const std::string & *name*, MessageHandler * *handler*, std::string *usage* = `""`, bool *reentrant* = `false`)

Registers the given Handler, by name, in the list of objects supported by the receiver. The name given is used to access the object from a Sender. If the reentrant flag is set, only a single instance of the Handler::process() method may be running at any given time.

The given handler will be deleted upon destruction of the Receiver.

9.699.3.2 bool Receiver::start (void)

Starts the receiver thread and suspends the caller until a message is received. Ideally, all Handler registration should be completed prior to calling start. This method will block indefinitely unless the `stop()` method is called from another thread.

9.699.3.3 void Receiver::stop (void)

Stops the receiver thread. This method will block until a lock on the Receiver can be acquired thus there is no guarantee of the point at which the stop will take effect with respect to incoming messages and/or outgoing replies.

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

- het_lib/messaging/include/receiver.h
- het_lib/messaging/src/receiver.cpp

9.700 tthread::recursive_mutex Class Reference

9.700.1 Detailed Description

Recursive mutex class. This is a mutual exclusion object for synchronizing access to shared memory areas for several threads. The mutex is recursive (i.e. a thread may lock the mutex several times, as long as it unlocks the mutex the same number of times).

See also:

mutex

`#include <tinythread.h>`

Public Member Functions

- recursive_mutex ()
  Constructor.
- ~recursive_mutex ()
  Destructor.
- void lock ()
- bool try_lock ()
- void unlock ()
- **recursive_mutex** (const recursive_mutex &)
- recursive_mutex & **operator=** (const recursive_mutex &)

Friends

- class **condition_variable**

9.700.2 Member Function Documentation

9.700.2.1 void tthread::recursive_mutex::lock () [inline]

Lock the mutex. The method will block the calling thread until a lock on the mutex can be obtained. The mutex remains locked until `unlock()` is called.

See also:

lock_guard

9.700.2.2 bool tthread::recursive_mutex::try_lock () [inline]

Try to lock the mutex. The method will try to lock the mutex. If it fails, the function will return immediately (non-blocking).

Returns:

`true` if the lock was acquired, or `false` if the lock could not be acquired.

9.700.2.3 void tthread::recursive_mutex::unlock () [inline]

Unlock the mutex. If any threads are waiting for the lock on this mutex, one of them will be unblocked.

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

- het_lib/tthread/include/tinythread.h

9.701 wfs_photometry::WFSCentroid::refList_t Struct Reference

Public Attributes

- double **X**
- double **Y**
- double **DX**
- double **DY**

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

- het_lib/wfs_lib/include/wfs_photometry.h

9.702 RelinquishControl Class Reference

9.702.1 Detailed Description

Relinquish pipeline control. The pipeline (camera) name, is prepended to the handler name (i.e. Guider1_).

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.703 Reset Class Reference

Reset the dome power inverter.

9.703.1 Detailed Description

Reset the shuttere power inverter.

Parameters:

none

`#include <legacyDomeSubSystem.h>`

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

- legacy/include/legacyDomeSubSystem.h

9.704 camra_handlers::reset Class Reference

Public Member Functions

- **reset** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.704.1 Member Function Documentation

9.704.1.1 void camra_handlers::reset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.705 reset Class Reference

9.705.1 Detailed Description

Resets the ARC hardware subsystem and rebuilds related wrapper classes.

XXX

`#include <camra_handler_doc.h>`

Public Member Functions

- **reset** (MetrologyLoop *l, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.705.2 Member Function Documentation

9.705.2.1 void reset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.706 camra_handlers::reset_clock_voltages Class Reference

Public Member Functions

- **reset_clock_voltages** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.706.1 Member Function Documentation

9.706.1.1 void camra_handlers::reset_clock_voltages::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.707 reset_clock_voltages Class Reference

9.707.1 Detailed Description

Resets the CCD clock voltages for all configured CCDs to the voltages given in the configuration.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.708 reset_focus_offset Class Reference

9.708.1 Detailed Description

Resets the current focus offset to zero.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **reset_focus_offset** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.708.2 Member Function Documentation

9.708.2.1 void reset_focus_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.709 reset_itf_offset Class Reference

9.709.1 Detailed Description

Resets the current ITF offset to to the base zero point offset.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **reset_itf_offset** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.709.2 Member Function Documentation

9.709.2.1 void reset_itf_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.710 camra_handlers::reset_mux Class Reference

Public Member Functions

- **reset_mux** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.710.1 Member Function Documentation

9.710.1.1 void camra_handlers::reset_mux::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.711 reset_mux Class Reference

9.711.1 Detailed Description

Resets the requested mux.

Required parameters:

index<int> - Index of the mux in the array, as enumerated by the software.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.712 reset_radec_offset Class Reference

9.712.1 Detailed Description

Resets the radec offset register to the current apparent coordinates.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **reset_radec_offset** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.712.2 Member Function Documentation

9.712.2.1 void reset_radec_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.713 reset_sky_offset Class Reference

9.713.1 Detailed Description

the current sky offset to zero.

Optional parameters:

move_probes<bool> - Flag indicating that the movable probes should be adjusted to account for reset of the offset.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **reset_sky_offset** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.713.2 Member Function Documentation

9.713.2.1 void reset_sky_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.714 camra_handlers::reset_tcs Class Reference

Public Member Functions

- **reset_tcs** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.714.1 Member Function Documentation

9.714.1.1 void camra_handlers::reset_tcs::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.715 reset_tcs Class Reference

9.715.1 Detailed Description

Flags the socket connection to tcs for resetting. The reset occurs on the next subsequent failed ACK from tcs.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.716 reset_tiptilt_offset Class Reference

9.716.1 Detailed Description

Resets the current tiptilt offset to zero.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **reset_tiptilt_offset** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.716.2 Member Function Documentation

9.716.2.1 void reset_tiptilt_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.717 RetractACQCameraMirror Class Reference

9.717.1 Detailed Description

Retract the Acquisition Camera mirror.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **RetractACQCameraMirror** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ACQCameraHandlers.cpp

9.718 RetractCWFSCameraMirror Class Reference

9.718.1 Detailed Description

Retract the Calibration Wave Front Sensor mirror.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **RetractCWFSCameraMirror** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/CwfsPvHandlers.cpp

9.719 ExposureInfo::ROI Struct Reference

Public Attributes

- int **x**
- int **y**
- int **width**
- int **height**

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

- pas/include/Camera.h

9.720 RowDispatcher Class Reference

9.720.1 Detailed Description

The RowDispatcher class manages the "job pool" for the fractal calculation threads.

Public Member Functions

- **RowDispatcher** (Image *aImage)
- Image * **GetImage** ()
- int **NextRow** ()

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

- het_lib/tthread/test/fractal.cpp

9.721 rtf_coordinate Class Reference

9.721.1 Detailed Description

Class representing a point in Real Tracker Frame coordinates, where coorections for mechanical mount have been applied.

`#include <tcs_coordinates.h>`

Public Member Functions

- **rtf_coordinate** (const fptime &t, const double &x, const double &y, const double &z, const double &rho, const double &theta, const double &phi)
- virtual void **apply_parms** (Payload &p) const

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

- het_lib/common/include/tcs_coordinates.h
- het_lib/common/src/tcs_coordinates.cpp

9.722 sampleConfig Class Reference

Public Member Functions

- **sampleConfig** (bool isNamed=0)
- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

- virtual void print_summary (void)

  Called prior to displaying options.

9.722.1 Member Function Documentation

9.722.1.1 virtual void sampleConfig::print_summary (void) [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.722.1.2 void sampleConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

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

- playground/sample_server/sampleConfig.h
- playground/sample_server/sampleConfig.cpp

9.723 SampleHandlerClass Class Reference

Public Member Functions

- **SampleHandlerClass** (TCSSubSystem *server)
- virtual void process (Message *pMessage)

9.723.1 Member Function Documentation

9.723.1.1 virtual void SampleHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements ServerHandler.

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

- playground/sample_server/sampleServer.h

9.724 sampleServer Class Reference

Public Member Functions

- **sampleServer** (const string &appName, sampleConfig *pConfig)
- virtual int **start** (void)
- virtual void **stop** (void)
- **HANDLER_DECL** (Sample)

Protected Member Functions

- void **registerReceiveMsgHandlers** (Receiver *pReceiver)
- void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)
- void **SetName** (string name)
- const char * **GetName** (void)
- void **SetDebug** (bool on)

Protected Attributes

- string **m_Name**
- bool **m_Debug**
- sampleConfig * **m_Config**
- Receiver * **m_Receiver**

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

- playground/sample_server/sampleServer.h
- playground/sample_server/sampleServer.cpp

9.725 pasDevicePipelineSubSystem::SAMSData Struct Reference

Public Member Functions

- **SAMSData** (SAMSData &other)
- bool **operator==** (SAMSData &rhs)
- bool **operator!=** (SAMSData &rhs)

Public Attributes

- double **GROCRemaining**
- double **RSE**
- double **targetRSE**
- double **tipTiltError**

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

- pas/include/pasDevicePipelineSubSystem.h

9.726 SAMSData Class Reference

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- struct SAMSData::status **m_status**

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

- gui/include/pasStatusData.h
- gui/src/pasStatusData.cpp

9.727 Sender Class Reference

9.727.1 Detailed Description

Sender class providing access to the Handler objects exposed by a Receiver.

`#include <sender.h>`

Public Types

- typedef threadsafe_deque< Message * > **message_queue**

Public Member Functions

- Sender (const std::string &endpoint, const std::string &name)
- Message * send (const std::string &handler, Message *message=NULL, unsigned timeout_secs=0)
- msgid_t send_async (std::string handler, Message *message=NULL)
- Message * wait_for (msgid_t id, bool do_exception)
- Message * **wait_for** (msgid_t id, bool do_exception, double timeout)
- message_queue * **get_message_queue** (void)
- void **set_message_queue** (message_queue *q)
- const std::string & **get_endpoint** ()
- void **reinitialize** (void)

Friends

- void **sender_recv_thread** (void *z, zctx_t *ctx, void *pipe)
- int **sender_shutdown_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **sender_ack_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **sender_done_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)
- int **sender_recv_handler** (zloop_t *loop, zmq_pollitem_t *item, void *z)

9.727.2 Constructor & Destructor Documentation

9.727.2.1 Sender::Sender (const std::string & *endpoint*, const std::string & *name*)

Sender constructor taking the `endpoint` of the Receiver instance to which it will connect, a `name` that will identify outgoing traffic, and an optional initial message that is incremented for each outgoing message. Applications requiring persistant serialization of the message `id` must manage the setting of this upon invocation.

9.727.3 Member Function Documentation

9.727.3.1 Message* Sender::send (const std::string & *handler*, Message * *message* = `NULL`, unsigned *timeout_secs* = `0`)

Sends the given `message` to the `Handler` object corresponding to the `handler` name given and ensures that the `message` is received and processed. Throws `MessagingException` if the message is not received or the `timeout_secs` have expired prior to the remote end completing the processing of the

message. If timeout_secs is zero, the method waits indefinitely for the remote end to complete the message processing.

Returns a pointer to the remote end's done response Message that must be deleted by the caller when it is no longer needed.

9.727.3.2 msgid_t Sender::send_async (std::string *handler*, Message * *message* = NULL)

Sends the given message to the Handler object corresponding to the handler name given and ensures that the message is received then returns to the caller. All responses from the remote end are pushed into the message queue. Throws MessagingException if the message is not received.

Returns an message ID useful for associating responses with a particular invocation. Returns MESSAGE_-INVALID if an error occurs.

9.727.3.3 Message * Sender::wait_for (msgid_t *id*, bool *do_exception*)

Waits for the "done" response to the given asynchronous message ID to arrive in the queue and then returns that message. If do_exception is true, and the response found is an error, the corresponding exception is raised.

The message returned should be deleted by the caller when it is no longer needed.

WARNING: This method is not very efficient.

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

- het_lib/messaging/include/sender.h
- het_lib/messaging/src/sender.cpp

9.728 ExposureInfo::Sensor Struct Reference

Public Attributes

- unsigned int **width**
- unsigned int **height**
- unsigned int **size**
- unsigned int **bitsPerPixel**
- unsigned int **bytesPerPixel**
- unsigned int **configWidth**
- unsigned int **configHeight**
- double **configPixelSize**
- unsigned int **wcsRefPixelX**
- unsigned int **wcsRefPixelY**
- int **gain**
- bool **tempSupported**
- double **temperature**
- bool **targetTempSupported**
- double **targetTemp**

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

- pas/include/Camera.h

9.729 ser7_table_entry Struct Reference

Public Member Functions

- **ser7_table_entry** (double dut, double pmx, double pmy)

Public Attributes

- double **dut_**
- double **pmx_**
- double **pmy_**

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

- het_lib/common/include/het_earth_orientation.h

9.730 Server Class Reference

9.730.1 Detailed Description

XXX

`#include <tracker_socket.h>`

Public Member Functions

- **Server** (int port)

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

- tracker/tracker_socket.h
- tracker/tracker_socket.cpp

9.731 ServerHandler Class Reference

Public Member Functions

- **ServerHandler** (const char ∗name, class TCSSubSystem ∗pServer)
- **ServerHandler** (const char ∗name, class TCSSubSystem ∗pServer, const MessageHandlerDescriptor &d)
- virtual void process (Message ∗message)=0

Protected Attributes

- const char ∗ **m_Name**
- TCSSubSystem ∗ **m_pServer**

9.731.1 Member Function Documentation

9.731.1.1 virtual void ServerHandler::process (Message ∗ *message*) [pure virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

Implemented in ForceStatusEventsHandlerClass, SampleHandlerClass, and ForceStatusEventsHandlerClass.

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

- het_lib/common/include/subsystem.h

9.732 set Class Reference

Public Member Functions

- **set** (Config &d, const MessageHandlerDescriptor &m)
- void process (Message *message)
- **set** (IglooSensors &d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.732.1 Member Function Documentation

9.732.1.1 void set::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

9.732.1.2 void set::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- ics/config.h
- igloo_sensors/igloo_sensors.h

9.733 set_analysis_region Class Reference

9.733.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **set_analysis_region** (GuideLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.733.2 Member Function Documentation

9.733.2.1 void set_analysis_region::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.734 camra_handlers::set_bias_offset Class Reference

Public Member Functions

- **set_bias_offset** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.734.1 Member Function Documentation

9.734.1.1 void camra_handlers::set_bias_offset::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.735 set_bias_offset Class Reference

9.735.1 Detailed Description

Set the bias offset of the given amplifier. Amplifier is specified as a combination of CCD identifier and position, either upper, lower, or all.

Required parameters:

count<int> - The offset count setting.

ccd<int> - Identifier of the CCD to operate on, defaults to 0 for all.

amp<string> - Amplifier to tune, either upper, lower, or all.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.736 camra_handlers::set_ccd_temp Class Reference

Public Member Functions

- **set_ccd_temp** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.736.1 Member Function Documentation

9.736.1.1 void camra_handlers::set_ccd_temp::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.737 set_ccd_temp Class Reference

9.737.1 Detailed Description

Set the target temperature setpoint of the given CCD.

Required parameters:

temp<double> - The target temperature setpoint.

Optional parameters:

ccd<int> - Identifier of the CCD to operate on, defaults to 0 for all.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.738 camra_handlers::set_ccd_temp_xfer Class Reference

Public Member Functions

- **set_ccd_temp_xfer** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.738.1 Member Function Documentation

9.738.1.1 void camra_handlers::set_ccd_temp_xfer::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.739 set_ccd_temp_xfer Class Reference

9.739.1 Detailed Description

Sets the parameters for the given CCDs temperature transfer function.

Required parameters:

ccd<int> - CCD ID to operate on.

bias<double> - Bias value for the transfer function.

gain<double> - Gain value for the transfer function.

```
#include <camra_handler_doc.h>
```

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

- camra/camra_handler_doc.h

9.740 camra_handlers::set_clock_voltages Class Reference

Public Member Functions

- **set_clock_voltages** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.740.1 Member Function Documentation

9.740.1.1 void camra_handlers::set_clock_voltages::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.741 set_clock_voltages Class Reference

9.741.1 Detailed Description

Sets the CCD clock voltages.

Optional parameters:

ccd<int> - Identifier of the CCD to operate on, defaults to 0 for all.

serial_1_high<double> - Sets the serial clock phase 1 high voltage to the given value.

serial_1_low<double> - Sets the serial clock phase 1 low voltage to the given value.

parallel_1_high<double> - Sets the parallel clock phase 1 high voltage to the given value.

parallel_1_low<double> - Sets the parallel clock phase 1 low voltage to the given value.

serial_2_high<double> - Sets the serial clock phase 2 high voltage to the given value.

serial_2_low<double> - Sets the serial clock phase 2 low voltage to the given value.

parallel_2_high<double> - Sets the parallel clock phase 2 high voltage to the given value.

parallel_2_low<double> - Sets the parallel clock phase 2 low voltage to the given value.

serial_3_high<double> - Sets the serial clock phase 3 high voltage to the given value.

serial_3_low<double> - Sets the serial clock phase 3 low voltage to the given value.

parallel_3_high<double> - Sets the parallel clock phase 3 high voltage to the given value.

parallel_3_low<double> - Sets the parallel clock phase 3 low voltage to the given value.

reset_gate_high<double> - Sets the reset gate high voltage to the given value.

reset_gate_low<double> - Sets the reset gate low voltage to the given value.

summing_well_high<double> - Sets the summing well high voltage to the given value.

summing_well_low<double> - Sets the summing well low voltage to the given value.

lower_output_drain<double> - Sets the output drain voltage for the lower amplifier.

upper_output_drain<double> - Sets the output drain voltage for the upper amplifier.

reset_drain<double> - Sets the reset drain voltage.

output_gate<double> - Sets the output gate voltage.

```
#include <camra_handler_doc.h>
```

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

- camra/camra_handler_doc.h

9.742 set_cryo_temp_xfer Class Reference

9.742.1 Detailed Description

Sets the parameters for the temperature transfer function for the cryostat RTD associated with the given controller ID.

Required parameters:

cntl<int> - Controller ID to operate on.

bias<double> - Bias value for the transfer function.

gain<double> - Gain value for the transfer function.

`#include <camra_handler_doc.h>`

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

- camra/camra_handler_doc.h

9.743 camra_handlers::set_cryo_temp_xfer Class Reference

Public Member Functions

- **set_cryo_temp_xfer** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.743.1 Member Function Documentation

9.743.1.1 void camra_handlers::set_cryo_temp_xfer::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- camra/camra_handlers.h

9.744 set_fiducial Class Reference

9.744.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **set_fiducial** (GuideLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.744.2 Member Function Documentation

9.744.2.1 void set_fiducial::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.745 set_filter Class Reference

9.745.1 Detailed Description

Set the requested filter for the camera.

Required parameters:

filter<int> - XXX update this to take a string... For now, integer representation of the desired filter.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **set_filter** (TCSDriver *d, PFIPClient &pfip, Instrument &instrument, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.745.2 Member Function Documentation

9.745.2.1 void set_filter::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/pfip_client.h

9.746 set_focus Class Reference

9.746.1 Detailed Description

Sets the focus of the system to a given w or the latest reading from the DMI.

Optional parameters:

w<double> - Focus setting in mm.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **set_focus** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.746.2 Member Function Documentation

9.746.2.1 void set_focus::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.747 set_focus_gain Class Reference

9.747.1 Detailed Description

Sets the gain factor applied to the focus correction emitted by the loop.

Optional parameters:

gain<double> - The gain by which the correction is scaled prior to being applied to the trajectory.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **set_focus_gain** (WFSLoop *l, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.747.2 Member Function Documentation

9.747.2.1 void set_focus_gain::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.748 set_focus_window_length Class Reference

9.748.1 Detailed Description

Sets the number of WFS focus measurements that are integrated to produce a single focus correction from the WFS loop.

Optional parameters:

length<unsigned> - Number of samples to integrate.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **set_focus_window_length** (WFSLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.748.2 Member Function Documentation

9.748.2.1 void set_focus_window_length::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.749 set_metadata Class Reference

9.749.1 Detailed Description

Sets all parameters given as header key metadata making the key/value pairs available via get_metadata.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **set_metadata** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.749.2 Member Function Documentation

9.749.2.1 void set_metadata::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/tcs_handlers.h

9.750 set_position Class Reference

9.750.1 Detailed Description

Set the target offset for the probe. This stages the move but does not execute probe moves. If parameters are given in x_ang, y_ang then the probe will be positioned at these telecentric angles off of the focal axis. Otherwise, ra and dec are expected and should be given relative to the trajectory at the time that the probes are moved. An equinox for the ra,dec is optional and defaults to 2000.

See also:

go_next, pfip_move_probes

Optional parameters:

x_ang<double> - On sky degrees off focal axis in X.

y_ang<double> - On sky degrees off focal axis in Y.

ra<double> - Right ascension in decimal hours.

dec<double> - Declination in degrees.

equinox<double> - Equinox for the given ra,dec.

trajectory<string> - If probe positions are given in absolute ra,dec they will be converted to focal plain positions relative to the ra,dec of the requested trajectory, either current or next. This defaults to current.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **set_position** (TCSDriver *d, PFIPClient &pfip, Instrument &instrument, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.750.2 Member Function Documentation

9.750.2.1 void set_position::process (Message * *message*) [virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/pfip_client.h
- tcs/pfip_client.cpp

9.751 set_tiptilt_gain Class Reference

9.751.1 Detailed Description

Sets the gain factor applied to the tiptilt correction emitted by the loop.

Optional parameters:

gain<double> - The gain by which the correction is scaled prior to being applied to the trajectory.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **set_tiptilt_gain** (WFSLoop *l, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.751.2 Member Function Documentation

9.751.2.1 void set_tiptilt_gain::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.752 set_tiptilt_window_length Class Reference

9.752.1 Detailed Description

Sets the number of WFS tiptilt measurements that are integrated to produce a single tiptilt correction from the WFS loop.

Optional parameters:

length<unsigned> - Number of samples to integrate.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **set_tiptilt_window_length** (WFSLoop *l, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.752.2 Member Function Documentation

9.752.2.1 void set_tiptilt_window_length::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tcs/metrology.h

9.753 SetACQCameraFilter Class Reference

9.753.1 Detailed Description

Change the acquisition camera filter setting

Required parameters:

pos<int> - An integer between 1 and 4 (inclusive) specifying the filter setting to use.

filter_name<string> - The configured name of a filter (e.g., \"i\", \"g\", \"B\").

`#include <pfip_handler_doc.h>`

Public Member Functions

- **SetACQCameraFilter** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ACQCameraHandlers.cpp

9.754 SetACQCameraFocus Class Reference

9.754.1 Detailed Description

Change the focus setting for the acquisition camera

Required parameters:

relative<float> - An number in units of milli-meters indicating the amount to change the focus setting. The valid focus setting is between 0 and 19.8.

absolute<float> - A absolute number in units of milli-meters indicating the new focus setting. The valid focus setting is between 0 and 19.8.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **SetACQCameraFocus** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ACQCameraHandlers.cpp

9.755 SetAnalysisRegion Class Reference

9.755.1 Detailed Description

Required parameters:

xmin<int> -

xmax<int> -

ymin<int> -

ymax<int> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.756 SetBinning Class Reference

Set camera binning.

9.756.1 Detailed Description

Required parameters:

width<int> -

height<int> -

Parameters:

width <int> - pixels per bin in rows.

height <int> - pixels per bin in columns.

```
#include <pas_handler_doc.h>
```

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

- pas/include/pas_handler_doc.h

9.757 setBufferSize Class Reference

9.757.1 Detailed Description

Sets the number of trajectory points that will be buffered prior to sending to TMCS.

Required parameters:

size<int> - Number of points

`#include <tracker_handler_doc.h>`

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

- tracker/tracker_handler_doc.h

9.758 tracker_handlers::setBufferSize Class Reference

Public Member Functions

- **setBufferSize** (TrackerDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.758.1 Member Function Documentation

9.758.1.1 void tracker_handlers::setBufferSize::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tracker/tracker_handlers.h

9.759 SetCleanImages Class Reference

9.759.1 Detailed Description

Required parameters:

clean<bool> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.760 SetCleaningBiasType Class Reference

9.760.1 Detailed Description

Required parameters:

type<string> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.761 SetCycleTime Class Reference

9.761.1 Detailed Description

Set the indicated pipeline's cycle speed. The pipeline (camera) name, is prepended to the handler name (i.e. Guider1_).

Required parameters:

seconds<double> - The cycle speed in seconds.

```
#include <pas_handler_doc.h>
```

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

- pas/include/pas_handler_doc.h

9.762 SetDirection Class Reference

Set the dome movement direction.

9.762.1 Detailed Description

Parameters:

direction <string> - movement direction ("CW", "CCW", "SHORTEST")

`#include <legacyDomeSubSystem.h>`

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

- legacy/include/legacyDomeSubSystem.h

9.763 SetENWAFilter Class Reference

9.763.1 Detailed Description

Change the ENWA (Entry Window) filter setting.

Required parameters:

pos<integer> - An integer between 1 and 3 (inclusive) that identifies the level of dithering.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **SetENWAFilter** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.764 SetExposuresPerCycle Class Reference

9.764.1 Detailed Description

Required parameters:

exposures<> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.765 SetExposureTime Class Reference

Set exposure time.

9.765.1 Detailed Description

Required parameters:

time<double> -

Parameters:

time <double> - time in ms

```
#include <pas_handler_doc.h>
```

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

- pas/include/pas_handler_doc.h

9.766 SetFilterExposureTimeFactor Class Reference

9.766.1 Detailed Description

Required parameters:

filterName<string> -

factor<double> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.767 SetFilterSequence Class Reference

9.767.1 Detailed Description

Required parameters:

sequence<string> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.768 SetFilterZeroPoint Class Reference

9.768.1 Detailed Description

Required parameters:

filterName<string> -

zeroPoint<double> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.769 SetGROC Class Reference

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

- pas/include/pas_handler_doc.h

9.770 SetGuideCameraFilter Class Reference

9.770.1 Detailed Description

Change the filter setting on a guider camera.

Required parameters:

guider<string> - Either 'Guider1' or 'Guider2', identifying the guider camera.

filter_name<string> - The configured name of a filter (e.g., \"i\", \"g\", \"Clear\").

pos<integer> - An integer between 1 and 5 (inclusive) specifying the filter setting for the guide camera.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **SetGuideCameraFilter** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.771 SetImageType Class Reference

Set image type.

9.771.1 Detailed Description

Required parameters:

type<string> -

Parameters:

type <string> - type of image ("OBJECT", "DARK", "BIAS")

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.772 SetIntervalSpeed Class Reference

Set the pipeline interval speed (time between images).

9.772.1 Detailed Description

Set the indicated pipeline's cycle time. The pipeline (camera) name, is prepended to the handler name (i.e. Guider1_).

Required parameters:

speed<double> - The cycle time in seconds.

Parameters:

***speed*<*int*>** - speed in ms

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.773 SetLightSourcePower Class Reference

9.773.1 Detailed Description

Required parameters:

state<bool> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.774 SetLightSourcePowerHoldTime Class Reference

9.774.1 Detailed Description

Required parameters:

time<> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.775 SetLightSourcePowerLevel Class Reference

9.775.1 Detailed Description

Required parameters:

level<double> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.776 SetNumFlushes Class Reference

Set number of flushes.

9.776.1 Detailed Description

Required parameters:

flushes<int> -

Parameters:

flushes <int> - Number of flushes (initial frames to discard)

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.777 SetObjectAndMagnitudes Class Reference

9.777.1 Detailed Description

Sets the observed object's name and magnitudes for each filter.

Required parameters:

object<string> - Identifies the observed object.

Optional parameters:

filterName<double> - The parameter name is one of the five filter names for the camera. The value is the magnitude associated with this filter. Up to five of these optional parameters may be specified.

```
#include <pas_handler_doc.h>
```

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

- pas/include/pas_handler_doc.h

9.778 SetOverScan Class Reference

9.778.1 Detailed Description

Required parameters:

over_scan<int> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.779 SetPhotPhotometryOptions Class Reference

9.779.1 Detailed Description

Required parameters:

mag_zero<double> -

detect_sigma<uble> -

min_pixels<int> -

max_pixels<int> -

nan_value<double> -

saturation<double> -

gain<double> -

sky_order<int> -

sky_annulus<int> -

kron_aperture<double> -

fixed_aperture<double> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.780 SetPixelFormat Class Reference

Set exposure type.

9.780.1 Detailed Description

Required parameters:

type<string> -

Parameters:

type <string> - type of exposure ("MONO_8", "MONO_16")

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.781 SetPower Class Reference

Set the dome power state.

9.781.1 Detailed Description

Set the shutter power state.

Parameters:

state <string> - state ("ON", "OFF")

`#include <legacyDomeSubSystem.h>`

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

- legacy/include/legacyDomeSubSystem.h

9.782 SetProbeGains Class Reference

9.782.1 Detailed Description

Set the gains for each probe.

Optional parameters:

probe_IGain<int> - Set the IGain for the probe identified by one of: 'Guider1', 'Guider2', 'WFS1', or 'WFS2'.

probe_PGain<int> - Set the PGain for the probe identified by one of: 'Guider1', 'Guider2', 'WFS1', or 'WFS2'.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **SetProbeGains** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

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

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.783 SetProcessImages Class Reference

9.783.1 Detailed Description

Required parameters:

process<bool> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.784 SetProcessingType Class Reference

9.784.1 Detailed Description

Required parameters:

type<string> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.785 SetReadoutSpeed Class Reference

Set readout speed.

9.785.1 Detailed Description

Required parameters:

speed<double> -

Parameters:

speed <string> - readout speed ("FAST", "SLOW")

```
#include <pas_handler_doc.h>
```

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

- pas/include/pas_handler_doc.h

9.786 SetRegionOfInterest Class Reference

Set sub-image rectangle.

9.786.1 Detailed Description

Required parameters:

xmin<int> -

xmax<int> -

ymin<int> -

ymax<> -

Parameters:

xmin <int> - xmin

xmax <int> - xmax

ymin <int> - ymin

ymax <int> - ymax

```
#include <pas_handler_doc.h>
```

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

- pas/include/pas_handler_doc.h

9.787 SetStoreImages Class Reference

9.787.1 Detailed Description

Required parameters:

store<bool> -

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.788 SetTargetTemperature Class Reference

Set target temperature.

9.788.1 Detailed Description

Required parameters:

temp<double> -

Parameters:

temp <double> - temperature (C)

`#include <pas_handler_doc.h>`

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

- pas/include/pas_handler_doc.h

9.789 setTraj Class Reference

9.789.1 Detailed Description

Sends a trajectory point to TMCS.

See also:

tracker_settraj, tmcs_set_traj

Required parameters:

id<int> - Identifier for the trajectory.

t<string> - ISO8601 formatted string.

x<double> - XXX

y<double> - XXX

z<double> - XXX

rho<double> - XXX

theta<double> - XXX

phi<double> - XXX

```
#include <tracker_handler_doc.h>
```

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

- tracker/tracker_handler_doc.h

9.790 tracker_handlers::setTraj Class Reference

Public Member Functions

- **setTraj** (TrackerDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.790.1 Member Function Documentation

9.790.1.1 void tracker_handlers::setTraj::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

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

- tracker/tracker_handlers.h

9.791 SetVelocity Class Reference

Set the dome movement velocity.

9.791.1 Detailed Description

Parameters:

velocity <double> - velocity in feet per second

`#include <legacyDomeSubSystem.h>`

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

- legacy/include/legacyDomeSubSystem.h

9.792 SetWFSPhotometryOptions Class Reference

9.792.1 Detailed Description

Required parameters:

bRenewFlag<bool> -

pixelsAcrossSubAperture<int> -

pixelsPerSubAperture<int> -

numZernikeModes<int> -

numZernikeModesToReconstruct<int> -

diskX<double> -

diskY<double> -

diskRadius<double> -

pixelScale<double> -

pupilRadius<double> -

rotation<double> -

refWaveLength<double> -

```
#include <pas_handler_doc.h>
```

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

- pas/include/pas_handler_doc.h

9.793 SHGrid Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual void **newData** (int size1, int size2)
- virtual void **deleteData** (void)
- void **operator=** (SHGrid &rhs)

Public Attributes

- ushort **Nsubap**
- ushort **Nvalid**
- ushort **Nvertx**
- ushort **shape**
- ushort **arrange**
- float **pitch**
- float **subapsize**
- float **x0**
- float **y0**
- float **psfdia**
- float **rotation**
- bool * **mask**
- float * **Xsubap_center**
- float * **Ysubap_center**
- float * **Xsubap_vertex**
- float * **Ysubap_vertex**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.794 ShotPoint Class Reference

Structure containing tracker variables; see HET TR 43 for details.

`#include <point.h>`

Public Attributes

- double dha_
  hours
- double beta_
  deg
- double ha_
  hours
- utc utc_
  broken down time

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

- integration/sir/Het_trajectory/point.h
- tcs/traj_comp/point.h

9.795 SHTrackerData Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual void **newData** (int size)
- virtual void **deleteData** (void)

Public Attributes

- ushort **Nelem**
- uint * **idx**
- double * **x**
- double * **y**
- double * **z**
- double * **p**

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

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.796 Shutter Class Reference

Public Member Functions

- **Shutter** (PFIPClient &pfip, const std::string &tty)
- void **enable** (void)
- void **disable** (void)
- void **actuate** (tthread::Timer &t, double duration, bool dark, bool subordinate)
- bool **open** (void)
- void **open** (double duration)
- bool **close** ()
- bool **active** (void)
- void **abort** ()
- void **wait_for_open** (volatile bool &override)
- bool **enabled** ()

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

- camra/shutter.h

9.797 shutter_Abort Class Reference

9.797.1 Detailed Description

Return shutter on-line status.

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.798 shutter_Close Class Reference

9.798.1 Detailed Description

Close the shutter.

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.799 shutter_IssuePMACCmd Class Reference

9.799.1 Detailed Description

Required parameters:

cmd<string> -

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.800 shutter_OnLine Class Reference

9.800.1 Detailed Description

Return shutter on-line status.

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.801 shutter_Open Class Reference

9.801.1 Detailed Description

Open the Shutter.

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.802 shutter_Reset Class Reference

9.802.1 Detailed Description

Reset the shutter power inverter.

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.803 shutter_SetPower Class Reference

9.803.1 Detailed Description

Required parameters:

state<string> -

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.804 shutter_Status Class Reference

9.804.1 Detailed Description

Optional parameters:

update_state<bool> -

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.805 AcqShutterStatus::shutter_status Struct Reference

Public Member Functions

- **UINT_VAR** (encoder, 6)
- **UINT_VAR** (encoderModulo, 6)
- **FPA_OPEN_CLOSED_VAR** (pos)

Public Attributes

- flags **m_flags**

The documentation for this struct was generated from the following file:

- gui/include/pfipStatusData.h

9.806 shutter_Stop Class Reference

9.806.1 Detailed Description

Stop (soft stop) shutter movement.

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.807 ShutterConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- string m_address

  Hardware or Legacy server address.

- double **m_moveTimeout**

9.807.1 Member Function Documentation

9.807.1.1 void ShutterConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

The documentation for this class was generated from the following files:

- legacy/include/legacyConfig.h
- legacy/src/legacyConfig.cpp

9.808 ShutterInfo Class Reference

Public Member Functions

- void **SetShutterStatus** (MAXONDriveStatus &s)
- void **SendShutterStatusEvent** ()
- void **SetEncoder** (uint Val, uint ModVal)
- void **SetPosition** (int pos)
- void **SetInternalState** (int state)
- int **GetShutterPos** ()
- void **SetShutterLastOpen** (const fptime &t)
- void **SetShutterLastClose** (const fptime &t)
- fptime & **GetShutterLastOpen** ()
- fptime & **GetShutterLastClose** ()
- void **SetSchedOpen** (const fptime &t)
- void **SetSchedClose** (const fptime &t)
- void **SetShutterDuration** (const fptime &t)
- fptime & **GetShutterDuration** ()

The documentation for this class was generated from the following files:

- pfip/include/PFIPStatus.h
- pfip/src/EventInfo.cpp

9.809 ShutterSetPowerHandlerClass Class Reference

Public Member Functions

- **ShutterSetPowerHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.809.1 Member Function Documentation

9.809.1.1 virtual void ShutterSetPowerHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

The documentation for this class was generated from the following file:

- legacy/include/legacyShutterSubSystem.h

9.810 ShutterStatus Class Reference

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- struct ShutterStatus::status **m_status**

The documentation for this class was generated from the following files:

- gui/include/legacyStatusData.h
- gui/src/legacyStatusData.cpp

9.811 ShutterStatusHandlerClass Class Reference

Public Member Functions

- **ShutterStatusHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.811.1 Member Function Documentation

9.811.1.1 virtual void ShutterStatusHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

The documentation for this class was generated from the following file:

- legacy/include/legacyShutterSubSystem.h

9.812 Ui::shutterTab Class Reference

The documentation for this class was generated from the following file:

- gui/build/ui_shutterTab.h

9.813 shutterTabUi Class Reference

Signals

- void **ForceHeartbeat** (std::string)
- void **ForceLegacyStatusEvents** (void)
- void **ForceLoopStateEvent** (std::string)

Public Member Functions

- **shutterTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

The documentation for this class was generated from the following files:

- gui/include/shutterTabUi.h
- gui/build/moc_shutterTabUi.cpp
- gui/src/shutterTabUi.cpp

9.814 SlopeMatrix Class Reference

Public Member Functions

- virtual void **importData** (const string &filename, format_t formatType, uchar *memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual int **exportData** (const string &filename, format_t formatType, uchar **memoryBlock=NULL, wfsData *additionalData=NULL)
- virtual void **newData** (int size1, int size2)
- virtual void **deleteData** (void)

Public Attributes

- ushort **nsubap**
- ushort **maxterm**
- gsl_matrix * **slope**

The documentation for this class was generated from the following files:

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.815 sockcmd_reg_id_t Struct Reference

9.815.1 Detailed Description

Information about the modbus register to be accessed by the communication thread

`#include <hwcomm.h>`

Public Attributes

- int id
- bool is_hld

9.815.2 Member Data Documentation

9.815.2.1 int sockcmd_reg_id_t::id

Integer identifier to be passed to libmodbus API

9.815.2.2 bool sockcmd_reg_id_t::is_hld

False to use read_input_registers(), true to use [read|write]_registers().

The documentation for this struct was generated from the following file:

- igloo_sensors/hwcomm.h

9.816 sockcmd_t Struct Reference

9.816.1 Detailed Description

Data payload of a socket message to comm_thread().

`#include <hwcomm.h>`

Public Attributes

- sockcmd_code_t cmd_code
- sockcmd_reg_id_t reg_id
- uint16_t value

9.816.2 Member Data Documentation

9.816.2.1 sockcmd_code_t sockcmd_t::cmd_code

Command code sent to thread, and response code returned.

9.816.2.2 sockcmd_reg_id_t sockcmd_t::reg_id

Register to be read or written.

9.816.2.3 uint16_t sockcmd_t::value

Value of register identified by reg_id, provided to a set operation, or returned from a get operation.

The documentation for this struct was generated from the following file:

- igloo_sensors/hwcomm.h

9.817 Socket Class Reference

9.817.1 Detailed Description

Socket class that can be inherited independent of the desired connection type (server/client). This class does not take care of reconnect and assumes that all errors are unrecoverable.

`#include <tracker_socket.h>`

Public Member Functions

- bool **create** ()
- bool **bind** (const int port)
- bool **listen** (const int max_connect=1)
- bool **accept** (Socket ∗new_socket)
- bool **connect** (const std::string host, const int port)
- bool **send** (const std::string(1.0 ∗(1.+78./6378.135)))
- int **recv** (std::string &(1.0 ∗(1.+78./6378.135)))
- const Socket & **operator**<< (const std::string &(1.0 ∗(1.+78./6378.135)))
- const Socket & **operator**>> (std::string &(1.0 ∗(1.+78./6378.135)))

The documentation for this class was generated from the following files:

- tracker/tracker_socket.h
- tracker/tracker_socket.cpp

9.818 SocketException Class Reference

9.818.1 Detailed Description

XXX

`#include <tracker_socket.h>`

Public Member Functions

- **SocketException** (const char *format,...)
- virtual const char * **what** () const throw ()

The documentation for this class was generated from the following file:

- tracker/tracker_socket.h

9.819 tcsGui::soundObject Class Reference

Public Attributes

- QString **m_soundName**
- bool **m_bRepeat**

The documentation for this class was generated from the following file:

- gui/include/tcsGui.h

9.820 tcsdb::source_descriptor Class Reference

Public Member Functions

- def **__init__**
- def **__call__**

Public Attributes

- **cursor_**
- **system_**
- **source_**

The documentation for this class was generated from the following file:

- scripting/pytcs/build/lib.linux-x86_64-2.7/tcsdb.py

9.821 pytcs::tcsdb::source_descriptor Class Reference

Public Member Functions

- def **__init__**
- def **__call__**

Public Attributes

- **cursor_**
- **system_**
- **source_**

The documentation for this class was generated from the following file:

- scripting/pytcs/tcsdb.py

9.822 hetlib::spectrograph::Spectrograph Class Reference

Public Member Functions

- def **__init__**
- def init_FmKeyStr
- def parse_KeyStr
- def Pad
- def **SpecIdStr**
- def **SsaIdStr**
- def **IfuSlotIdStr**
- def **CntlIdStr**
- def **MuxIdStr**
- def **CryoPressStr**
- def **CryoTempStr**
- def **CcdTempStr**
- def **CcdSetPtStr**
- def **CcdDeltaStr**
- def **CcdIdStr**
- def **HeaterStr**
- def PowerStr
- def **CcdBiasStr**
- def mkDisplay
- def **make_Temperature_Command**
- def **__init__**
- def init_FmKeyStr
- def parse_KeyStr
- def Pad
- def **SpecIdStr**
- def **SsaIdStr**
- def **IfuSlotIdStr**
- def **CntlIdStr**
- def **MuxIdStr**
- def **CryoPressStr**
- def **CryoTempStr**
- def **CcdTempStr**
- def **CcdSetPtStr**
- def **CcdDeltaStr**
- def **CcdIdStr**
- def **HeaterStr**
- def PowerStr
- def **CcdBiasStr**
- def mkDisplay
- def **make_Temperature_Command**

9.822.1 Member Function Documentation

9.822.1.1 def hetlib::spectrograph::Spectrograph::init_FmKeyStr (*self*, *initstr*)

```
Initialize the dictionary with the basic information,
instrument, spectrograph, mux, and controller from the
hardware status key. The key has the form "virus.mux.000.spec.020.cntl.088...."
```

9.822.1.2 def hetlib::spectrograph::Spectrograph::init_FmKeyStr (*self*, *initstr*)

```
Initialize the dictionary with the basic information,
instrument, spectrograph, mux, and controller from the
hardware status key. The key has the form "virus.mux.000.spec.020.cntl.088...."
```

9.822.1.3 def hetlib::spectrograph::Spectrograph::mkDisplay (*self*, *display* = `'ALL'`, *strtype* = `'data'`, *indent* = `0`, *datetime* = `None`)

```
Print the values in a spectro object. Items printed are determined
by the display variables. Options are ALL, TEMPS, CRYOTEMPS, CCDTEMPS, PRESSURE, or HEATER.
```

9.822.1.4 def hetlib::spectrograph::Spectrograph::mkDisplay (*self*, *display* = `'ALL'`, *strtype* = `'data'`, *indent* = `0`, *datetime* = `None`)

```
Print the values in a spectro object. Items printed are determined
by the display variables. Options are ALL, TEMPS, CRYOTEMPS, CCDTEMPS, PRESSURE, or HEATER.
```

9.822.1.5 def hetlib::spectrograph::Spectrograph::Pad (*self*, *count* = `0`, *padchar* = `' '`)

```
Pad strings with some characters, the default is a space character
```

9.822.1.6 def hetlib::spectrograph::Spectrograph::Pad (*self*, *count* = `0`, *padchar* = `' '`)

```
Pad strings with some characters, the default is a space character
```

9.822.1.7 def hetlib::spectrograph::Spectrograph::parse_KeyStr (*self*, *parsestr*, *val*)

```
Fill the dictionary with the specific information from a
hardware status (key, val). The key has the form
"'ccd.right.15369.amp.upper.bias': 3885". Returns True/False if you
care about whether the string was parsed.
```

9.822.1.8 def hetlib::spectrograph::Spectrograph::parse_KeyStr (*self*, *parsestr*, *val*)

```
Fill the dictionary with the specific information from a
hardware status (key, val). The key has the form
"'ccd.right.15369.amp.upper.bias': 3885". Returns True/False if you
care about whether the string was parsed.
```

9.822.1.9 def hetlib::spectrograph::Spectrograph::PowerStr (*self*, *strtype*, *ind* = `0`)

```
From the equation P=IE and E=IR, we can derive that P=(E^2)/R. Since
100% power occurs at E = 1.733134 then the power percentage for a specific
heater voltage is P/P_100 = (E^2/r) / (1.733134^2 /R) or P = 100.0 * (E/1.733134)^2
```

9.822.1.10 def hetlib::spectrograph::Spectrograph::PowerStr (*self*, *strtype*, *ind* = 0)

```
From the equation P=IE and E=IR, we can derive that P=(E^2)/R. Since
100% power occurs at E = 1.733134 then the power percentage for a specific
heater voltage is P/P_100 = (E^2/r) / (1.733134^2 /R) or P = 100.0 * (E/1.733134)^2
```

The documentation for this class was generated from the following files:

- scripting/build/lib/hetlib/spectrograph.py
- scripting/hetlib/spectrograph.py

9.823 SpectrographConfig Class Reference

Spectrograph configuration class.

9.823.1 Detailed Description

Supports spectrograph specific custom parameterization of the CCD array.

`#include <camra_config.h>`

Public Member Functions

- void set_options (OptionParser &o)

  Adds options to default set.

- void print_summary (void)

  Called prior to displaying options.

Public Attributes

- int controller_id

  Serial number of the controller.

- int **characterization_controller_id**
- int spectrograph_id
- int **spectrograph_slot**
- string **ifu_id**
- int **ifu_slot**
- int ccd_left_id

  Serial numbers of the CCDs.

- int **ccd_right_id**
- float **ccd_left_temp_setpoint**
- float **ccd_right_temp_setpoint**
- string **pressure_gauge**
- float **ccd_left_upper_gain**
- float **ccd_left_lower_gain**
- float **ccd_right_upper_gain**
- float **ccd_right_lower_gain**
- float **ccd_left_upper_rdnoise**
- float **ccd_left_lower_rdnoise**
- float **ccd_right_upper_rdnoise**
- float **ccd_right_lower_rdnoise**
- int **ccd_left_upper_bias**
- int **ccd_left_lower_bias**
- int **ccd_right_upper_bias**
- int **ccd_right_lower_bias**
- int **ccd_left_bias_target**

- int **ccd_right_bias_target**
- std::map< string, float > **ccd_right_voltages**
- std::map< string, float > **ccd_left_voltages**
- bool **ignore_clock_voltages**
- int **ccd_left_amp_config**
- int **ccd_right_amp_config**
- ControllerConfig * **cntl_config**

9.823.2 Member Function Documentation

9.823.2.1 void SpectrographConfig::print_summary (void) [virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the -h output.

Implements TCSConfig.

9.823.2.2 void SpectrographConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

9.823.3 Member Data Documentation

9.823.3.1 int SpectrographConfig::spectrograph_id

Identifiers and relative locations of the imaging hardware. These are initialized as configured and, if not in "lab mode" vetted against the as discovered hardware information.

The documentation for this class was generated from the following files:

- camra/camra_config.h
- camra/camra_config.cpp

9.824 hetlib::spectrograph::SpectrographList Class Reference

9.824.1 Detailed Description

```
A list of Spectrograph objects and things you can do with them
```

Public Member Functions

- def **__init__**
- def find
- def sortSpect
- def **get_mux_from_log**
- def **__init__**
- def find
- def sortSpect
- def **get_mux_from_log**

9.824.2 Member Function Documentation

9.824.2.1 def hetlib::spectrograph::SpectrographList::find (*self*, *spectId* = ″, *cntlId* = ″, *ccdId* = ″)

```
return spectro object
```

9.824.2.2 def hetlib::spectrograph::SpectrographList::find (*self*, *spectId* = ″, *cntlId* = ″, *ccdId* = ″)

```
return spectro object
```

9.824.2.3 def hetlib::spectrograph::SpectrographList::sortSpect (*self*, *sortBy* = 'IFU', *reverse* = False)

```
Sorting Spectrographs only makes sense if there is
a single entry.
```

9.824.2.4 def hetlib::spectrograph::SpectrographList::sortSpect (*self*, *sortBy* = 'IFU', *reverse* = False)

```
Sorting Spectrographs only makes sense if there is
a single entry.
```

The documentation for this class was generated from the following files:

- scripting/build/lib/hetlib/spectrograph.py
- scripting/hetlib/spectrograph.py

9.825 HETillumination::Spidermotion Struct Reference

Public Attributes

- float **dx2dtheta_itf**
- float **dx2dphi_itf**
- float **dy2dtheta_itf**
- float **dy2dphi_itf**
- float **drho2drho_itf**
- float **dx2dtel_x**
- float **dx2dtel_y**
- float **dy2dtel_x**
- float **dy2dtel_y**
- float **drho2dcarriage**
- float **drho2darm**

The documentation for this struct was generated from the following file:

- het_lib/illum_lib/include/illumination.h

9.826 fixed_point::fixed_internal::square< N > Struct Template Reference

template<unsigned int N> struct fixed_point::fixed_internal::square< N >

Public Types

- enum { **value** = N∗N }

The documentation for this struct was generated from the following file:

- het_lib/common/include/fixed.h

9.827 start Class Reference

9.827.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **start** (MetrologyLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.827.2 Member Function Documentation

9.827.2.1 void start::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/metrology.h

9.828 StartPipeline Class Reference

Start the pipleine for a particular camera.

9.828.1 Detailed Description

Start cycling the indicated pipeline. The pipeline (camera) name, is prepended to the handler name (i.e. Guider1_).

Parameters:

none

`#include <pas_handler_doc.h>`

The documentation for this class was generated from the following file:

- pas/include/pas_handler_doc.h

9.829 StartSendingImages Class Reference

Start sending images to the sender of this message.

9.829.1 Detailed Description

Parameters:

none

`#include <pasImagePipelineSubSystem.h>`

The documentation for this class was generated from the following file:

- pas/include/pas_handler_doc.h

9.830 StaticConfig Struct Reference

Public Member Functions

- **StaticConfig** (std::string pupilgeom, std::string gridgeom, std::string imagemask, std::string modalcube, std::string trkergeom, std::string wfcm4geom, std::string spidrgeom, std::string m1geom, std::string calibdata, float pix2mm, float pix2rad, float wavelen, float dmi0, float wfsclock, float TT2xyoffset, float DF2woffset, float CM2tpoffset, int xSkyOffsetSign, int ySkyOffsetSign, int wFocusOffsetSign, int thetaOffsetSign, int phiOffsetSign, float M1xmotion2itf_theta, float M1xmotion2itf_phi, float M1ymotion2itf_theta, float M1ymotion2itf_phi, float TRKxmotion2itf_theta, float TRKxmotion2itf_phi, float TRKymotion2itf_theta, float TRKymotion2itf_phi, float ROTmotion2itf_rho, float ROTmotion2prb_crg, float ROTmotion2prb_arm, float M4motion2prb_telx, float M4motion2prb_tely, float xSkyOffsetCalibration, float ySkyOffsetCalibration, float wFocusOffsetCalibration, float thetaOffsetCalibration, float phiOffsetCalibration, float PRBcrganglezero, float PRBarmanglezero, float wfs_gain, float wfs_flen, CentroidInfo centroidoptions)
- **StaticConfig** (std::string pupilgeom, std::string gridgeom, std::string imagemask, std::string modalcube, std::string trkergeom, std::string wfcm4geom, std::string spidrgeom, std::string m1geom, std::string calibdata, float pix2mm, float pix2rad, float wavelen, float dmi0, float wfsclock, float TT2xyoffset, float DF2woffset, float CM2tpoffset, int xSkyOffsetSign, int ySkyOffsetSign, int wFocusOffsetSign, int thetaOffsetSign, int phiOffsetSign, float M1xmotion2itf_theta, float M1xmotion2itf_phi, float M1ymotion2itf_theta, float M1ymotion2itf_phi, float TRKxmotion2itf_theta, float TRKxmotion2itf_phi, float TRKymotion2itf_theta, float TRKymotion2itf_phi, float ROTmotion2itf_rho, float ROTmotion2prb_crg, float ROTmotion2prb_arm, float M4motion2prb_telx, float M4motion2prb_tely, float xSkyOffsetCalibration, float ySkyOffsetCalibration, float wFocusOffsetCalibration, float thetaOffsetCalibration, float phiOffsetCalibration, float PRBcrganglezero, float PRBarmanglezero, float wfs_gain, float wfs_flen, CentroidInfo centroidoptions)

Public Attributes

- bool **complete**
- std::string **m_pupilgeom**
- std::string **m_gridgeom**
- std::string **m_imagemask**
- std::string **m_modalcube**
- std::string **m_trkergeom**
- std::string **m_wfcm4geom**
- std::string **m_spidrgeom**
- std::string **m_m1geom**
- std::string **m_calibdata**
- float **m_pix2mm**
- float **m_pix2rad**
- float **m_wavelen**
- float **m_dmi0**
- float **m_wfsclock**
- float **m_TT2xyoffset**
- float **m_DF2woffset**
- float **m_CM2tpoffset**
- int **m_xSkyOffsetSign**
- int **m_ySkyOffsetSign**
- int **m_wFocusOffsetSign**
- int **m_thetaOffsetSign**

- int **m_phiOffsetSign**
- float **m_M1xmotion2itf_theta**
- float **m_M1xmotion2itf_phi**
- float **m_M1ymotion2itf_theta**
- float **m_M1ymotion2itf_phi**
- float **m_TRKxmotion2itf_theta**
- float **m_TRKxmotion2itf_phi**
- float **m_TRKymotion2itf_theta**
- float **m_TRKymotion2itf_phi**
- float **m_ROTmotion2itf_rho**
- float **m_ROTmotion2prb_crg**
- float **m_ROTmotion2prb_arm**
- float **m_M4motion2prb_telx**
- float **m_M4motion2prb_tely**
- float **m_xSkyOffsetCalibration**
- float **m_ySkyOffsetCalibration**
- float **m_wFocusOffsetCalibration**
- float **m_thetaOffsetCalibration**
- float **m_phiOffsetCalibration**
- float **m_PRBcrganglezero**
- float **m_PRBarmanglezero**
- float **m_wfs_gain**
- float **m_wfs_flen**
- CentroidInfo **m_centroidoptions**
- float **MINIMUM_SUBAP_SNR**
- float **MAXIMUM_PSF_RADIUS**
- float **HET_MM2ASEC**
- float **WFS_PLATE_SCALE**

The documentation for this struct was generated from the following files:

- het_lib/wfs_lib/include/reducewfs.h
- het_lib/wfs_lib/include/wfslib_het.h

9.831 legacyWeatherSubSystem::statistic Struct Reference

Public Attributes

- string **sPrefix**
- string **sSuffix**
- double **dValue**
- bool **bValid**

The documentation for this struct was generated from the following file:

- legacy/include/legacyWeatherSubSystem.h

9.832 WeatherData::statistic Struct Reference

Public Attributes

- QString **m_label**
- double **m_value**

The documentation for this struct was generated from the following file:

- gui/include/legacyStatusData.h

9.833 PipelineStatus::status Struct Reference

Public Member Functions

- **INT_VAR** (cameraType, 2)
- **DOUBLE_VAR** (targetTemp, 6, 1)
- **DOUBLE_VAR** (cameraTemp, 6, 1)
- **DOUBLE_VAR** (electronicsTemp, 6, 1)

The documentation for this struct was generated from the following file:

- gui/include/pipelineStatusData.h

9.834 DomeStatus::status Struct Reference

Public Member Functions

- **BOOL_VAR** (valid)
- **DOUBLE_VAR** (position, 6, 2)
- **DOUBLE_VAR** (target, 6, 2)
- **DOUBLE_VAR** (velocity, 4, 1)
- **DOUBLE_VAR** (maxVelocity, 4, 1)
- **STRING_VAR** (direction)
- **BOOL_VAR** (moving)
- **BOOL_VAR** (PMACComms)
- **INT_VAR** (errorNum, 5)
- **STRING_VAR** (localRemote)
- **BOOL_VAR** (EMO)

The documentation for this struct was generated from the following file:

- gui/include/legacyStatusData.h

9.835 Status Class Reference

Return the dome status information.

9.835.1 Detailed Description

Return structure status information.

Return the status status information.

Parameters:

none

`#include <legacyDomeSubSystem.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacyDomeSubSystem.h

9.836 WeatherData::status Struct Reference

Public Member Functions

- **STRING_VAR** (GMT)
- **STRING_VAR** (currentWarnings)
- **STRING_VAR** (masterWarnings)
- **DOUBLE_VAR_ARRAY** (temp, 7, 2, 4)
- **DOUBLE_VAR_ARRAY** (humidity, 7, 2, 4)
- **DOUBLE_VAR_ARRAY** (intDewPoint, 7, 2, 4)
- **DOUBLE_VAR_ARRAY** (extDewPoint, 7, 2, 4)
- **DOUBLE_VAR_ARRAY** (pressure, 6, 2, 4)
- **DOUBLE_VAR_ARRAY** (windDir, 7, 2, 4)
- **DOUBLE_VAR_ARRAY** (windSpeed, 7, 2, 4)
- **DOUBLE_VAR_ARRAY** (extDust1um, 11, 2, 4)
- **DOUBLE_VAR_ARRAY** (extDustPt3um, 11, 2, 4)
- **DOUBLE_VAR_ARRAY** (intDust1um, 11, 2, 4)
- **DOUBLE_VAR_ARRAY** (intDustPt3um, 11, 2, 4)
- **DOUBLE_VAR_ARRAY** (rain, 6, 2, 4)

The documentation for this struct was generated from the following file:

- gui/include/legacyStatusData.h

9.837 SAMSData::status Struct Reference

Public Member Functions

- **BOOL_VAR** (pipelineRunning)
- **DOUBLE_VAR** (cycleTime, 9, 3)
- **DOUBLE_VAR** (RSE, 8, 3)
- **DOUBLE_VAR** (targetRSE, 8, 3)
- **DOUBLE_VAR** (tipTiltError, 5, 3)

The documentation for this struct was generated from the following file:

- gui/include/pasStatusData.h

9.838 legacyWeatherSubSystem::status Struct Reference

Public Member Functions

- **status** (status &other)
- bool **operator==** (status &rhs)
- bool **operator!=** (status &rhs)

Public Attributes

- bool **bValid**
- string **sGMT**
- string **sCurrentWarnings**
- string **sMasterWarnings**
- statistic **temp** [4]
- statistic **humidity** [4]
- statistic **intDewPoint** [4]
- statistic **extDewPoint** [4]
- statistic **pressure** [4]
- statistic **windDir** [4]
- statistic **windSpeed** [4]
- statistic **extDust1** [4]
- statistic **extDust2** [4]
- statistic **intDust1** [4]
- statistic **intDust2** [4]
- statistic **rain** [4]
- statistic **truss** [4]
- statistic **trussAir** [4]
- statistic **tracker** [4]
- statistic **mirror** [4]

The documentation for this struct was generated from the following file:

- legacy/include/legacyWeatherSubSystem.h

9.839 PFIPStatusBase::status Struct Reference

Public Member Functions

- **STRING_VAR** (probeName)
- **INT_VAR** (probeEncoderVal, 6)
- **DOUBLE_VAR** (probeAngleDeg, 6, 3)
- **DOUBLE_VAR** (telecenX, 7, 4)
- **DOUBLE_VAR** (telecenY, 7, 4)
- **INT_VAR** (armEncoderVal, 6)
- **DOUBLE_VAR** (armAngleDeg, 6, 3)
- **DOUBLE_VAR** (armTelecenDeg, 6, 3)
- **DOUBLE_VAR** (probeOrientation, 6, 3)
- **INT_VAR** (filter, 2)
- **STRING_VAR** (filterNames)
- **STRING_VAR** (currentPos)

Public Attributes

- flags **m_probeFlags**
- flags **m_armFlags**
- flags **m_filterFlags**

The documentation for this struct was generated from the following file:

- gui/include/pfipStatusData.h

9.840 APCData::status Struct Reference

The documentation for this struct was generated from the following file:

- gui/include/apcStatusData.h

9.841 TrackerStatus::status Struct Reference

Public Member Functions

- QString **idx_time** (void)
- QString **ux_tsf_mm** (void)
- QString **lx_tsf_mm** (void)
- QString **y_tsf_mm** (void)
- QString **rho_tsf_deg** (void)
- QString **hex1_tsf_mm** (void)
- QString **hex2_tsf_mm** (void)
- QString **hex3_tsf_mm** (void)
- QString **hex4_tsf_mm** (void)
- QString **hex5_tsf_mm** (void)
- QString **hex6_tsf_mm** (void)
- QString **rho_abs_deg** (void)
- QString **ux_abs_mm** (void)
- QString **lx_abs_mm** (void)
- QString **y_abs_mm** (void)
- QString **cfd_load_cell** (void)
- QString **cfd_speed** (void)
- QString **ux_track_motor_current** (void)
- QString **ux_slew_motor_current** (void)
- QString **lx_track_motor_current** (void)
- QString **lx_slew_motor_current** (void)
- QString **y_track_motor_current** (void)
- QString **cfd_motor_current** (void)
- QString **y_slew_motor_current** (void)
- QString **hex1_motor_current** (void)
- QString **hex2_motor_current** (void)
- QString **hex3_motor_current** (void)
- QString **hex4_motor_current** (void)
- QString **hex5_motor_current** (void)
- QString **hex6_motor_current** (void)
- QString **rho_motor_current** (void)
- QString **ux_track_cmd_rpms** (void)
- QString **ux_slew_cmd_rpms** (void)
- QString **lx_track_cmd_rpms** (void)
- QString **lx_slew_cmd_rpms** (void)
- QString **y_track_cmd_rpms** (void)
- QString **y_slew_cmd_rpms** (void)
- QString **hex1_cmd_amps** (void)
- QString **hex2_cmd_amps** (void)
- QString **hex3_cmd_amps** (void)
- QString **hex4_cmd_amps** (void)
- QString **hex5_cmd_amps** (void)
- QString **hex6_cmd_amps** (void)
- QString **rho_cmd_rpms** (void)
- QString **x_track_lock_out** (void)
- QString **x_slew_lock_out** (void)

- QString **y_lock_out** (void)
- QString **hex1_lock_out** (void)
- QString **hex2_lock_out** (void)
- QString **hex3_lock_out** (void)
- QString **hex4_lock_out** (void)
- QString **hex5_lock_out** (void)
- QString **hex6_lock_out** (void)
- QString **rho_lock_out** (void)
- QString **emergency_stop** (void)
- QString **lx_left_soft_limit_sw** (void)
- QString **lx_right_soft_limit_sw** (void)
- QString **ux_left_soft_limit_sw** (void)
- QString **ux_right_soft_limit_sw** (void)
- QString **y_top_soft_limit_sw** (void)
- QString **y_bottom_soft_limit_sw** (void)
- QString **rho_soft_limit_sw** (void)
- QString **fiber_left_soft_limit_sw** (void)
- QString **fiber_right_soft_limit_sw** (void)
- QString **lx_left_hard_limit_sw** (void)
- QString **lx_right_hard_limit_sw** (void)
- QString **ux_left_hard_limit_sw** (void)
- QString **ux_right_hard_limit_sw** (void)
- QString **y_top_hard_limit_sw** (void)
- QString **y_bottom_hard_limit_sw** (void)
- QString **hex1_top_hard_limit_sw** (void)
- QString **hex1_bottom_hard_limit_sw** (void)
- QString **hex2_top_hard_limit_sw** (void)
- QString **hex2_bottom_hard_limit_sw** (void)
- QString **hex3_top_hard_limit_sw** (void)
- QString **hex3_bottom_hard_limit_sw** (void)
- QString **hex4_top_hard_limit_sw** (void)
- QString **hex4_bottom_hard_limit_sw** (void)
- QString **hex5_top_hard_limit_sw** (void)
- QString **hex5_bottom_hard_limit_sw** (void)
- QString **hex6_top_hard_limit_sw** (void)
- QString **hex6_bottom_hard_limit_sw** (void)
- QString **rho_cw_hard_limit_sw** (void)
- QString **rho_ccw_hard_limit_sw** (void)
- QString **lx_slew_brake_pressure** (void)
- QString **lx_track_brake_pressure** (void)
- QString **ux_slew_brake_pressure** (void)
- QString **ux_track_brake_pressure** (void)
- QString **y_slew_brake_pressure** (void)
- QString **y_track_brake_pressure** (void)
- QString **lx_skew_sensor_1** (void)
- QString **lx_skew_sensor_2** (void)
- QString **ux_skew_sensor_1** (void)
- QString **ux_skew_sensor_2** (void)
- QString **ux_track_servo** (void)
- QString **ux_slew_servo** (void)

- QString **lx_track_servo** (void)
- QString **lx_slew_servo** (void)
- QString **y_track_servo** (void)
- QString **cfd_servo** (void)
- QString **y_slew_servo** (void)
- QString **hex1_servo** (void)
- QString **hex2_servo** (void)
- QString **hex3_servo** (void)
- QString **hex4_servo** (void)
- QString **hex5_servo** (void)
- QString **hex6_servo** (void)
- QString **rho_servo** (void)
- QString **servo_cabinet_line_monitor** (void)
- QString **hex1_tsf_cmd_mm** (void)
- QString **hex2_tsf_cmd_mm** (void)
- QString **hex3_tsf_cmd_mm** (void)
- QString **hex4_tsf_cmd_mm** (void)
- QString **hex5_tsf_cmd_mm** (void)
- QString **hex6_tsf_cmd_mm** (void)
- QString **ux_tsf_cmd_mm** (void)
- QString **lx_tsf_cmd_mm** (void)
- QString **y_tsf_cmd_mm** (void)
- QString **rho_tsf_cmd_deg** (void)
- QString **x_rtf_mm** (void)
- QString **y_rtf_mm** (void)
- QString **z_rtf_mm** (void)
- QString **rho_rtf_deg** (void)
- QString **theta_rtf_deg** (void)
- QString **phi_rtf_deg** (void)
- QString **op_mode** (void)
- QString **trajectory** (void)
- QString **abort** (void)
- QString **moving** (void)
- QString **initing** (void)
- QString **slewing** (void)
- QString **tracking** (void)

Public Attributes

- double **m_idx_time**
- double **m_ux_tsf_mm**
- double **m_lx_tsf_mm**
- double **m_y_tsf_mm**
- double **m_rho_tsf_deg**
- double **m_hex1_tsf_mm**
- double **m_hex2_tsf_mm**
- double **m_hex3_tsf_mm**
- double **m_hex4_tsf_mm**
- double **m_hex5_tsf_mm**
- double **m_hex6_tsf_mm**

- double **m_rho_abs_deg**
- double **m_ux_abs_mm**
- double **m_lx_abs_mm**
- double **m_y_abs_mm**
- double **m_cfd_load_cell**
- double **m_cfd_speed**
- double **m_ux_track_motor_current**
- double **m_ux_slew_motor_current**
- double **m_lx_track_motor_current**
- double **m_lx_slew_motor_current**
- double **m_y_track_motor_current**
- double **m_cfd_motor_current**
- double **m_y_slew_motor_current**
- double **m_hex1_motor_current**
- double **m_hex2_motor_current**
- double **m_hex3_motor_current**
- double **m_hex4_motor_current**
- double **m_hex5_motor_current**
- double **m_hex6_motor_current**
- double **m_rho_motor_current**
- double **m_ux_track_cmd_rpms**
- double **m_ux_slew_cmd_rpms**
- double **m_lx_track_cmd_rpms**
- double **m_lx_slew_cmd_rpms**
- double **m_y_track_cmd_rpms**
- double **m_y_slew_cmd_rpms**
- double **m_hex1_cmd_amps**
- double **m_hex2_cmd_amps**
- double **m_hex3_cmd_amps**
- double **m_hex4_cmd_amps**
- double **m_hex5_cmd_amps**
- double **m_hex6_cmd_amps**
- double **m_rho_cmd_rpms**
- bool **m_x_track_lock_out**
- bool **m_x_slew_lock_out**
- bool **m_y_lock_out**
- bool **m_hex1_lock_out**
- bool **m_hex2_lock_out**
- bool **m_hex3_lock_out**
- bool **m_hex4_lock_out**
- bool **m_hex5_lock_out**
- bool **m_hex6_lock_out**
- bool **m_rho_lock_out**
- bool **m_emergency_stop**
- bool **m_lx_left_soft_limit_sw**
- bool **m_lx_right_soft_limit_sw**
- bool **m_ux_left_soft_limit_sw**
- bool **m_ux_right_soft_limit_sw**
- bool **m_y_top_soft_limit_sw**
- bool **m_y_bottom_soft_limit_sw**

- bool **m_rho_soft_limit_sw**
- bool **m_fiber_left_soft_limit_sw**
- bool **m_fiber_right_soft_limit_sw**
- bool **m_lx_left_hard_limit_sw**
- bool **m_lx_right_hard_limit_sw**
- bool **m_ux_left_hard_limit_sw**
- bool **m_ux_right_hard_limit_sw**
- bool **m_y_top_hard_limit_sw**
- bool **m_y_bottom_hard_limit_sw**
- bool **m_hex1_top_hard_limit_sw**
- bool **m_hex1_bottom_hard_limit_sw**
- bool **m_hex2_top_hard_limit_sw**
- bool **m_hex2_bottom_hard_limit_sw**
- bool **m_hex3_top_hard_limit_sw**
- bool **m_hex3_bottom_hard_limit_sw**
- bool **m_hex4_top_hard_limit_sw**
- bool **m_hex4_bottom_hard_limit_sw**
- bool **m_hex5_top_hard_limit_sw**
- bool **m_hex5_bottom_hard_limit_sw**
- bool **m_hex6_top_hard_limit_sw**
- bool **m_hex6_bottom_hard_limit_sw**
- bool **m_rho_cw_hard_limit_sw**
- bool **m_rho_ccw_hard_limit_sw**
- bool **m_lx_slew_brake_pressure**
- bool **m_lx_track_brake_pressure**
- bool **m_ux_slew_brake_pressure**
- bool **m_ux_track_brake_pressure**
- bool **m_y_slew_brake_pressure**
- bool **m_y_track_brake_pressure**
- bool **m_lx_skew_sensor_1**
- bool **m_lx_skew_sensor_2**
- bool **m_ux_skew_sensor_1**
- bool **m_ux_skew_sensor_2**
- bool **m_ux_track_servo**
- bool **m_ux_slew_servo**
- bool **m_lx_track_servo**
- bool **m_lx_slew_servo**
- bool **m_y_track_servo**
- bool **m_cfd_servo**
- bool **m_y_slew_servo**
- bool **m_hex1_servo**
- bool **m_hex2_servo**
- bool **m_hex3_servo**
- bool **m_hex4_servo**
- bool **m_hex5_servo**
- bool **m_hex6_servo**
- bool **m_rho_servo**
- bool **m_servo_cabinet_line_monitor**
- double **m_hex1_tsf_cmd_mm**
- double **m_hex2_tsf_cmd_mm**

- double **m_hex3_tsf_cmd_mm**
- double **m_hex4_tsf_cmd_mm**
- double **m_hex5_tsf_cmd_mm**
- double **m_hex6_tsf_cmd_mm**
- double **m_ux_tsf_cmd_mm**
- double **m_lx_tsf_cmd_mm**
- double **m_y_tsf_cmd_mm**
- double **m_rho_tsf_cmd_deg**
- double **m_x_rtf_mm**
- double **m_y_rtf_mm**
- double **m_z_rtf_mm**
- double **m_rho_rtf_deg**
- double **m_theta_rtf_deg**
- double **m_phi_rtf_deg**
- int **m_op_mode**
- bool **m_trajectory**
- bool **m_abort**
- bool **m_moving**
- bool **m_initing**
- bool **m_slewing**
- bool **m_tracking**

The documentation for this struct was generated from the following file:

- gui/include/trackerStatusData.h

9.842 OCDStatus::status Struct Reference

Public Member Functions

- **BOOL_VAR** (metrologyStateGood)
- **INT_VAR** (runShotState, 1)

The documentation for this struct was generated from the following file:

- gui/include/ocdStatusData.h

9.843 ThermoCubeStatus::status Struct Reference

Public Member Functions

- QString **ComputedSetpoint** (void)
- QString **DomeTempActual** (void)
- QString **DomeTempAverage** (void)
- QString **GlycolReturnTempActual** (void)
- QString **GlycolReturnTempAverage** (void)
- QString **GlycolSupplyTempActual** (void)
- QString **GlycolSupplyTempAverage** (void)
- QString **OffsetCurrent** (void)
- QString **ThermoCubeCurrentSetpoint** (void)
- QString **ThermoCubeCurrentTemp** (void)
- QString **VirusEnclTempActual** (void)
- QString **VirusEnclTempAverage** (void)
- QString **ControllingTemps** (void)
- QString **FanFailure** (void)
- QString **NoFaults** (void)
- QString **PumpFailure** (void)
- QString **RTDOpen** (void)
- QString **RTDShort** (void)
- QString **Running** (void)
- QString **TankLevelLow** (void)
- QString **ThermoCubeIsCooling** (void)

Public Attributes

- double **m_ComputedSetpoint**
- double **m_DomeTempActual**
- double **m_DomeTempAverage**
- double **m_GlycolReturnTempActual**
- double **m_GlycolReturnTempAverage**
- double **m_GlycolSupplyTempActual**
- double **m_GlycolSupplyTempAverage**
- double **m_OffsetCurrent**
- double **m_ThermoCubeCurrentSetpoint**
- double **m_ThermoCubeCurrentTemp**
- double **m_VirusEnclTempActual**
- double **m_VirusEnclTempAverage**
- bool **m_ControllingTemps**
- bool **m_FanFailure**
- bool **m_NoFaults**
- bool **m_PumpFailure**
- bool **m_RTDOpen**
- bool **m_RTDShort**
- bool **m_Running**
- bool **m_TankLevelLow**
- bool **m_ThermoCubeIsCooling**

The documentation for this struct was generated from the following file:

- gui/include/thermoCubeStatusData.h

9.844 ShutterStatus::status Struct Reference

Public Member Functions

- **BOOL_VAR** (valid)
- **STRING_VAR** (position)
- **STRING_VAR** (motion)
- **STRING_VAR** (target)
- **BOOL_VAR** (moving)
- **BOOL_VAR** (PMACComms)
- **INT_VAR** (errorNum, 5)
- **STRING_VAR** (localRemote)

The documentation for this struct was generated from the following file:

- gui/include/legacyStatusData.h

9.845 TrackerPosition::status Struct Reference

Public Member Functions

- **DOUBLE_VAR** (x_itf_mm, 9, 3)
- **DOUBLE_VAR** (y_itf_mm, 9, 3)
- **DOUBLE_VAR** (z_itf_mm, 9, 3)
- **DOUBLE_VAR** (rho_itf_deg, 6, 2)
- **DOUBLE_VAR** (theta_itf_deg, 6, 2)
- **DOUBLE_VAR** (phi_itf_deg, 6, 2)
- **DOUBLE_VAR** (x_rtf_mm, 9, 3)
- **DOUBLE_VAR** (y_rtf_mm, 9, 3)
- **DOUBLE_VAR** (z_rtf_mm, 9, 3)
- **DOUBLE_VAR** (rho_rtf_deg, 6, 2)
- **DOUBLE_VAR** (theta_rtf_deg, 6, 2)
- **DOUBLE_VAR** (phi_rtf_deg, 6, 2)
- **DOUBLE_VAR** (ux_tsf_mm, 9, 3)
- **DOUBLE_VAR** (lx_tsf_mm, 9, 3)
- **DOUBLE_VAR** (y_tsf_mm, 9, 3)
- **DOUBLE_VAR** (rho_tsf_deg, 6, 2)
- **DOUBLE_VAR** (hex1_tsf_mm, 9, 3)
- **DOUBLE_VAR** (hex2_tsf_mm, 9, 3)
- **DOUBLE_VAR** (hex3_tsf_mm, 9, 3)
- **DOUBLE_VAR** (hex4_tsf_mm, 9, 3)
- **DOUBLE_VAR** (hex5_tsf_mm, 9, 3)
- **DOUBLE_VAR** (hex6_tsf_mm, 9, 3)
- **BOOL_VAR** (x_motion)
- **BOOL_VAR** (y_motion)
- **BOOL_VAR** (rho_motion)
- **BOOL_VAR** (hex1_motion)
- **BOOL_VAR** (hex2_motion)
- **BOOL_VAR** (hex3_motion)
- **BOOL_VAR** (hex4_motion)
- **BOOL_VAR** (hex5_motion)
- **BOOL_VAR** (hex6_motion)

The documentation for this struct was generated from the following file:

- gui/include/trackerPositionData.h

9.846 legacyShutterSubSystem::status Struct Reference

Public Member Functions

- **status** (status &other)
- bool **operator==** (status &rhs)
- bool **operator!=** (status &rhs)

Public Attributes

- bool **bValid**
- bool **bActive**
- string **activeHandler**
- string **sPosition**
- string **sMotion**
- string **sTarget**
- bool **bMoving**
- bool **bPMACComms**
- int **nErrorNum**
- string **sLocalRemote**

The documentation for this struct was generated from the following file:

- legacy/include/legacyShutterSubSystem.h

9.847 VirusMonitorStatus::status Struct Reference

Public Member Functions

- **DOUBLE_VAR** (PurgeAirTemp, 6, 2)
- **DOUBLE_VAR** (LowerEENCTemp, 6, 2)
- **DOUBLE_VAR** (FarUpperVENCTemp, 6, 2)
- **DOUBLE_VAR** (FarLowerVENCTemp, 6, 2)
- **DOUBLE_VAR** (PurgeAirDewPointTemp, 6, 2)
- **DOUBLE_VAR** (PurgeAirFlow, 6, 2)
- **DOUBLE_VAR** (VENCPressure, 6, 2)
- **DOUBLE_VAR** (HEPAPressure, 6, 2)
- **DOUBLE_VAR** (VENCHeatExchangerInletTemp, 6, 2)
- **DOUBLE_VAR** (VENCHeatExchangerOutletTemp, 6, 2)
- **BOOL_VAR** (PurgeAirValve)

The documentation for this struct was generated from the following file:

- gui/include/pfipStatusData.h

9.848 PFIPMonitorStatus::status Struct Reference

Public Member Functions

- **DOUBLE_VAR** (DomeTemp, 6, 2)
- **DOUBLE_VAR** (LowerTemp, 6, 2)
- **DOUBLE_VAR** (UpperTemp, 6, 2)
- **DOUBLE_VAR** (FPATemp, 6, 2)
- **DOUBLE_VAR** (FPAPurgeAirTemp, 6, 2)
- **DOUBLE_VAR** (FPAGlycolInTemp, 6, 2)
- **DOUBLE_VAR** (FPAGlycolOutTemp, 6, 2)
- **DOUBLE_VAR** (FPAPurgeAirDewPointTemp, 6, 2)
- **DOUBLE_VAR** (FPAPurgeAirFlow, 6, 2)
- **DOUBLE_VAR** (WFCTemp, 6, 2)
- **DOUBLE_VAR** (WFCPurgeGN2Temp, 6, 2)
- **DOUBLE_VAR** (WFCPurgeGN2DewPointTemp, 6, 2)
- **DOUBLE_VAR** (WFCPurgeGN2Flow, 6, 2)

The documentation for this struct was generated from the following file:

- gui/include/pfipStatusData.h

9.849 DMIData::status Struct Reference

Public Member Functions

- **BOOL_VAR** (pipelineRunning)
- **DOUBLE_VAR** (cycleTime, 9, 3)
- **DOUBLE_VAR** (ccasDistance, 9, 6)
- **DOUBLE_VAR** (ccasPeakAmpl, 9, 6)
- **DOUBLE_VAR** (ccasCoupl1, 9, 6)
- **DOUBLE_VAR** (sacDistance, 9, 6)
- **DOUBLE_VAR** (sacPeakAmpl, 9, 6)
- **DOUBLE_VAR** (sacCoupl1, 9, 6)

The documentation for this struct was generated from the following file:

- gui/include/pasStatusData.h

9.850 StructureStatus::status Struct Reference

Public Member Functions

- **BOOL_VAR** (valid)
- **DOUBLE_VAR** (azimuth, 9, 4)
- **DOUBLE_VAR** (target, 9, 4)
- **BOOL_VAR** (moving)
- **BOOL_VAR** (inControl)
- **DOUBLE_VAR_ARRAY_CONCAT** (footHeight, 6, 2, 8)

The documentation for this struct was generated from the following file:

- gui/include/legacyStatusData.h

9.851 legacyDomeSubSystem::status Struct Reference

Public Member Functions

- **status** (status &other)
- bool **operator==** (status &rhs)
- bool **operator!=** (status &rhs)

Public Attributes

- bool **bValid**
- bool **bActive**
- string **activeHandler**
- double **dPosition**
- double **dTarget**
- double **dVelocity**
- double **dMaxVelocity**
- string **sDirection**
- bool **bMoving**
- bool **bPMACComms**
- int **nErrorNum**
- string **sLocalRemote**
- bool **bEMO**

The documentation for this struct was generated from the following file:

- legacy/include/legacyDomeSubSystem.h

9.852 IglooData::status Struct Reference

Public Member Functions

- **BOOL_VAR** (connectOk)
- **BOOL_VAR** (powerEnabled)
- **BOOL_VAR** (coolantEnabled)
- **BOOL_VAR** (fcPowerOn)
- **BOOL_VAR** (coolantShutoffDisable)

The documentation for this struct was generated from the following file:

- gui/include/iglooStatusData.h

9.853 legacyStructureSubSystem::status Struct Reference

Public Member Functions

- **status** (status &other)
- bool **operator==** (status &rhs)
- bool **operator!=** (status &rhs)
- status & **operator=** (const status &rhs)

Public Attributes

- bool **bValid**
- bool **bActive**
- string **activeHandler**
- double **dTrueAzimuth**
- double **dStructAzimuth**
- double **dTarget**
- double **dElevation**
- bool **bNewFault**
- bool **bInPosition**
- bool **bInControl**
- bool **bMoving**
- double **dFootHeight** [8]

The documentation for this struct was generated from the following file:

- legacy/include/legacyStructureSubSystem.h

9.854 StatusBlock< T > Class Template Reference

template<class T> class StatusBlock< T >

Public Attributes

- tthread::mutex **m_AccessMutex**
- fptime **m_LastStatusNotIgnoredAt**
- T * **m_pData**

The documentation for this class was generated from the following files:

- gui/include/tcsGui.h
- utility_gui/include/utilityGui.h

9.855 StatusData Class Reference

Public Member Functions

- **StatusData** (const char ∗name)
- virtual bool **initialize** (Payload &payload, void ∗pPrevStatus=NULL)=0

The documentation for this class was generated from the following file:

- gui/include/statusData.h

9.856 StatusHandler Class Reference

9.856.1 Detailed Description

Replies with a list of the currently active handler invocations.

XXX what else?

`#include <builtin_handlers.h>`

Public Member Functions

- virtual void process (Message ∗message)

9.856.2 Member Function Documentation

9.856.2.1 virtual void StatusHandler::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- het_lib/messaging/include/builtin_handlers.h

9.857 stencil< T > Class Template Reference

template<class T> class stencil< T >

Public Member Functions

- MArray< T, 2 > **getCircle** (const int radius) const

The documentation for this class was generated from the following file:

- het_lib/photometry/src/psf_zernickes.cpp

9.858 Stop Class Reference

Stop the current dome command (normally movement).

9.858.1 Detailed Description

Stop the structure move and set the structure down on the pier.

Stop the current shutter command (normally movement).

Parameters:

none

`#include <legacyDomeSubSystem.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacyDomeSubSystem.h

9.859 stop Class Reference

9.859.1 Detailed Description

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **stop** (MetrologyLoop ∗l, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.859.2 Member Function Documentation

9.859.2.1 void stop::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/metrology.h

9.860 StopPipeline Class Reference

Stop the pipleine for a particular camera.

9.860.1 Detailed Description

Stop cycling the indicated pipeline. The pipeline (camera) name, is prepended to the handler name (i.e. Guider1_).

Parameters:

none

`#include <pas_handler_doc.h>`

The documentation for this class was generated from the following file:

- pas/include/pas_handler_doc.h

9.861 StopSendingImages Class Reference

Stop sending images to the sender of this message.

9.861.1 Detailed Description

Parameters:

none

`#include <pasImagePipelineSubSystem.h>`

The documentation for this class was generated from the following file:

- pas/include/pas_handler_doc.h

9.862 Stow Class Reference

Stow the structure.

9.862.1 Detailed Description

Parameters:

none

`#include <legacyStructureSubSystem.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacyStructureSubSystem.h

9.863 structure_MoveToCCAS Class Reference

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.864 structure_MoveToStructAz Class Reference

9.864.1 Detailed Description

Required parameters:

target<double> -

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.865 structure_MoveToTrueAz Class Reference

9.865.1 Detailed Description

Required parameters:

target<double> -

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.866 structure_OnLine Class Reference

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.867 structure_Status Class Reference

9.867.1 Detailed Description

Optional parameters:

update_state<string> -

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.868 structure_Stop Class Reference

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.869 structure_Stow Class Reference

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.870 StructureClient Class Reference

Public Member Functions

- **StructureClient** (const std::string &route)
- bool **get_azimuth_and_elevation** (double &az, double &el, bool do_exception=0)
- void **update_status** (Event *event)
- void **move** (double az)
- void **wait** (void)

The documentation for this class was generated from the following files:

- tcs/structure_client.h
- tcs/structure_client.cpp

9.871 StructureConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)
  Adds options to default set.

Public Attributes

- int m_connectRetryCount
  number of connect retries
- bool m_bUseFootCommand
  Use FOOT vs. FOOT? command.
- string m_modelFilename
  Path name of mount model file.
- double m_cmdInactivityTimeout
  Command response inactivity timeout in seconds.
- double m_moveStartedTimeout
  Move started timeout in seconds.
- double m_moveFinishedTimeout
  Move finished timeout in seconds.
- double m_moveSetDownTimeout
  Move set down timeout in seconds.
- double m_positionAccuracyDegrees
  On target position accuracy.
- double m_noMoveNeededDistanceDegrees
  No move needed distance.

9.871.1 Member Function Documentation

9.871.1.1 void StructureConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

The documentation for this class was generated from the following files:

- legacy/include/legacyConfig.h
- legacy/src/legacyConfig.cpp

9.872 StructureModel Class Reference

Legacy Mount Model.

`#include <structureModel.h>`

Public Member Functions

- bool **ReadCoefficients** (std::string fileName)
- double **StructToTrueAz** (double dStructAz)
- double **StructToTrueEl** (double dTrueAz)
- double **TrueToStructAz** (double dTrueAz)

Static Public Member Functions

- static StructureModel & **Instance** (void)

Protected Types

- enum { **STRUCTURE_AZIMUTH_AXIS** = 1, **STRUCTURE_ELEVATION_AXIS** }

Protected Member Functions

- void **initModel** (void)
- bool **getNextLine** (std::ifstream &filePtr, char *buf, int len, char comChar= '#')
- double **applyModel** (int nAxis, double dAz, double dEl)
- double **dResiduals** (int nAxis, double dAz)
- int **applyModelRev** (double dXp, double dYp, double *dX, double *dY)

Protected Attributes

- double **m_AzimuthOffset_rad**
- double **m_ElevationOffset_rad**
- int **m_nNumCoefs**
- int **m_nNumResiduals**
- double **m_dCoefs** [6]
- double **m_dResiduals** [3][100]

The documentation for this class was generated from the following files:

- het_lib/coord_trans/include/structureModel.h
- het_lib/coord_trans/src/structureModel.cpp

9.873 StructureMoveToCCASHandlerClass Class Reference

Public Member Functions

- **StructureMoveToCCASHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.873.1 Member Function Documentation

9.873.1.1 virtual void StructureMoveToCCASHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

The documentation for this class was generated from the following file:

- legacy/include/legacyStructureSubSystem.h

9.874 StructureMoveToStructAzHandlerClass Class Reference

Public Member Functions

- **StructureMoveToStructAzHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.874.1 Member Function Documentation

9.874.1.1 virtual void StructureMoveToStructAzHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

The documentation for this class was generated from the following file:

- legacy/include/legacyStructureSubSystem.h

9.875 StructureMoveToTrueAzHandlerClass Class Reference

Public Member Functions

- **StructureMoveToTrueAzHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.875.1 Member Function Documentation

9.875.1.1 virtual void StructureMoveToTrueAzHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

The documentation for this class was generated from the following file:

- legacy/include/legacyStructureSubSystem.h

9.876 StructureOnLineHandlerClass Class Reference

Public Member Functions

- **StructureOnLineHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.876.1 Member Function Documentation

9.876.1.1 virtual void StructureOnLineHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

The documentation for this class was generated from the following file:

- legacy/include/legacyStructureSubSystem.h

9.877 StructureStatus Class Reference

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- struct StructureStatus::status **m_status**

The documentation for this class was generated from the following files:

- gui/include/legacyStatusData.h
- gui/src/legacyStatusData.cpp

9.878 StructureStatusHandlerClass Class Reference

Public Member Functions

- **StructureStatusHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.878.1 Member Function Documentation

9.878.1.1 virtual void StructureStatusHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

The documentation for this class was generated from the following file:

- legacy/include/legacyStructureSubSystem.h

9.879 StructureStowHandlerClass Class Reference

Public Member Functions

- **StructureStowHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.879.1 Member Function Documentation

9.879.1.1 virtual void StructureStowHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

The documentation for this class was generated from the following file:

- legacy/include/legacyStructureSubSystem.h

9.880 Ui::structureTab Class Reference

The documentation for this class was generated from the following file:

- gui/build/ui_structureTab.h

9.881 structureTabUi Class Reference

Signals

- void **ForceHeartbeat** (std::string)
- void **ForceLegacyStatusEvents** (void)
- void **ForceLoopStateEvent** (std::string)
- void **MoveStructureToTrueAz** (double)
- void **MoveStructureToStructAz** (double)

Public Member Functions

- **structureTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Protected Member Functions

- void **LoadVelocityValues** (void)

The documentation for this class was generated from the following files:

- gui/include/structureTabUi.h
- gui/build/moc_structureTabUi.cpp
- gui/src/structureTabUI.cpp

9.882 SubSystem Class Reference

Public Member Functions

- **SubSystem** (SubSystemConfig *pConfig)
- void **notifyStatusChanged** (void)
- virtual void **registerReceiveMsgHandlers** (Receiver *pReceiver)=0
- virtual void **unregisterReceiveMsgHandlers** (Receiver *pReceiver)=0
- virtual void **start** (bool takeControl=false, bool checkSocketConnection=false)
- virtual void **stop** (bool relinquishControl=false)
- virtual bool **initialize** (void)
- virtual void **waitComplete** (void)
- virtual void **stopAndWaitComplete** (void)
- virtual void **setPipelineCycleTime_ms** (int cycleTime_ms)
- virtual int **getPipelineCycleTime_ms** (void)
- virtual void **setStatusThreadCycleTime_ms** (int cycleTime_ms)
- virtual int **getStatusThreadCycleTime_ms** (void)
- virtual void **setStatusThreadForceEventCycleTime_ms** (int cycleTime_ms)
- virtual int **getStatusThreadForceEventCycleTime_ms** (void)
- virtual void **populateStatusPayload** (Payload &)
- virtual bool **checkConnection** (bool &bValid)
- virtual bool **isConnected** (void)
- virtual void **forceStatusEvent** (void)
- virtual void **clearActive** (Message *)
- **SubSystem** (const string &appName, Config *config)
- int **start** ()
- void **stop** ()
- bool **init** ()

Protected Member Functions

- virtual bool **attemptReconnect** (void)
- virtual void **startStatusWorkerThread** (bool checkSocketConnection=true)
- virtual void **statusWorkerPumpThreadEntryPoint** (void)
- virtual double **statusWorker** (bool=false)
- virtual bool **sendCmd** (int cmd, char *cmdString, bool(*checkDone)(unsigned char *, unsigned int, unsigned int), const char *terminators="\n", unsigned int timeout_sec=5)
- virtual bool **checkReply** (int, unsigned int, unsigned char *)
- virtual bool **getStatus** (bool retry=false, bool doException=false, bool extraBool=false)
- virtual void **checkAndSetActive** (Message *)

Protected Attributes

- tthread::mutex **m_sendCmdMutex**
- tthread::mutex **m_statusMutex**
- int **m_pipelineCycleTime_ms**
- int **m_statusThreadCycleTime_ms**
- int **m_statusThreadForceEventCycleTime_ms**
- tthread::thread * **m_pStatusThread**

- volatile bool **m_bStatusThreadCheckSocketConnection**
- volatile bool **m_bStatusThreadRunning**
- volatile bool **m_bStopStatusThread**
- tthread::condition_variable **m_forceStatusEvent**
- bool **m_bForceEventIgnoreChanges**

The documentation for this class was generated from the following files:

- het_lib/common/include/subsystem.h
- playground/toy_subsystem/toy_subsystem.cpp
- het_lib/common/src/subsystem.cpp

9.883 subSystem Class Reference

Public Member Functions

- **subSystem** (const string &name)
- virtual bool **init** (void)
- virtual int **start** (void)=0
- virtual void **stop** (void)

Public Attributes

- const string & **m_name**

The documentation for this class was generated from the following file:

- tcsconsole/tcsevents.cpp

9.884 liborchestrate::Subsystem Class Reference

Public Member Functions

- def **__init__**
- def **getClient**
- def **shutdown**
- def **__del__**

Public Attributes

- **pid**
- **name**
- **clients**

The documentation for this class was generated from the following file:

- integration/liborchestrate.py

9.885 SubSystemConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)
  Adds options to default set.
- virtual void print_summary (void)
  Called prior to displaying options.

Public Attributes

- string m_address
  Hardware or Legacy server address.
- string m_type
  Generic type.
- bool m_disabled
  Enable the subsystem.
- double m_statusThreadCycleTime_secs
  Status thread cycle time.
- double m_statusThreadForceEventCycle_secs
  Status thread force event cycle time.
- bool m_socketDebug
  Enable socket debugging.

9.885.1 Member Function Documentation

9.885.1.1 virtual void SubSystemConfig::print_summary (void) [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.885.1.2 void SubSystemConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

Reimplemented in pduConfig, DomeConfig, ShutterConfig, StructureConfig, WeatherConfig, LightSourceConfig, CameraConfig, CleanerConfig, ProcessorConfig, DeviceConfig, and PipelineConfig.

The documentation for this class was generated from the following files:

- het_lib/common/include/subsystem.h
- het_lib/common/src/subsystem.cpp

9.886 SubSystemHandler Class Reference

Public Member Functions

- **SubSystemHandler** (const char *name, class SubSystem *pSubSystem)
- **SubSystemHandler** (const char *name, class SubSystem *pSubSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *message)=0

Protected Attributes

- const char * **m_Name**
- SubSystem * **m_pSubSystem**

9.886.1 Member Function Documentation

9.886.1.1 virtual void SubSystemHandler::process (Message * *message*) [pure virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

Implemented in DomeStatusHandlerClass, ShutterSetPowerHandlerClass, ShutterStatusHandlerClass, StructureOnLineHandlerClass, StructureMoveToTrueAzHandlerClass, StructureMoveToStructAzHandlerClass, StructureMoveToCCASHandlerClass, StructureStatusHandlerClass, StructureStowHandlerClass, WeatherOnLineHandlerClass, WeatherGetDataHandlerClass, DevicePipelineSetGROCHandlerClass, DevicePipelineStatusHandlerClass, DevicePipelineEnableSACDMIHandlerClass, DevicePipelineEnableCCASDMIHandlerClass, ImagePipelineStartSendingImagesHandlerClass, ImagePipelineGetRegionOfInterestHandlerClass, ImagePipelineGetExposureTimeHandlerClass, ImagePipelineSetOverScanHandlerClass, ImagePipelineGetOverScanHandlerClass, ImagePipelineSetFilterExposureTimeFactorHandlerClass, ImagePipelineSetFilterSequenceHandlerClass, ImagePipelineSetFilterZeroPointHandlerClass, ImagePipelineGetGainHandlerClass, ImagePipelineSetObjectAndMagnitudesHandlerClass, ImagePipelineSetCleanImagesHandlerClass, ImagePipelineSetCleaningBiasTypeHandlerClass, ImagePipelineSetProcessingTypeHandlerClass, ImagePipelineGetPhotPhotometryOptionsHandlerClass, ImagePipelineGetWFSPhotometryOptionsHandlerClass, ImagePipelineSetPhotometryOptionsHandlerClass, ImagePipelineSetPhotPhotometryOptionsHandlerClass, ImagePipelineSetWFSPhotometryOptionsHandlerClass, PipelineSetCycleTimeHandlerClass, PipelineSetIntervalSpeedHandlerClass, PipelineStartPipelineHandlerClass, PipelineTakeControlHandlerClass, PipelineRelinquishControlHandlerClass, and PipelineCyclePipelineHandlerClass.

The documentation for this class was generated from the following file:

- het_lib/common/include/subsystem.h

9.887 liborchestrate::SubsystemManager Class Reference

Public Member Functions

- def **__init__**
- def **add**
- def **shutdown**

Public Attributes

- **systems**

The documentation for this class was generated from the following file:

- integration/liborchestrate.py

9.888 pytcs::tcsdb::system_descriptor Class Reference

Public Member Functions

- def **__init__**
- def **__call__**

Public Attributes

- **cursor_**
- **system_**

The documentation for this class was generated from the following file:

- scripting/pytcs/tcsdb.py

9.889 tcsdb::system_descriptor Class Reference

Public Member Functions

- def **__init__**
- def **__call__**

Public Attributes

- **cursor_**
- **system_**

The documentation for this class was generated from the following file:

- scripting/pytcs/build/lib.linux-x86_64-2.7/tcsdb.py

9.890 apcTabUi::tabData Class Reference

Public Attributes

- int **m_layoutRow**
- int **m_layoutCol**
- QGridLayout ∗ **m_gridLayout**

The documentation for this class was generated from the following file:

- gui/include/apcTabUi.h

9.891 TakeControl Class Reference

9.891.1 Detailed Description

Take control of pipeline. The pipeline (camera) name, is prepended to the handler name (i.e. Guider1_).

`#include <pas_handler_doc.h>`

The documentation for this class was generated from the following file:

- pas/include/pas_handler_doc.h

9.892 TakeExposure Class Reference

9.892.1 Detailed Description

Open the focal plabe camera shutter for a specified duration.

Required parameters:

duration<float> - Number of seconds to leave the shutter open.

pos<integer> - An integer between 1 and 5 (inclusive) specifying the filter setting for the guide camera.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **TakeExposure** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

The documentation for this class was generated from the following files:

- pfip/include/pfipServer.h
- pfip/src/ACQShutterHandlers.cpp

9.893 Tap Class Reference

Public Member Functions

- void **toggle** (void)
- bool **on** (void)
- void **recv** (Message *m)
- void **send** (Message *m)

The documentation for this class was generated from the following files:

- het_lib/messaging/include/tap.h
- het_lib/messaging/src/tap.cpp

9.894 TapHandler Class Reference

Public Member Functions

- virtual void process (Message *message)

9.894.1 Member Function Documentation

9.894.1.1 virtual void TapHandler::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- het_lib/messaging/include/builtin_handlers.h

9.895 TCSApp< Subsys, Config > Class Template Reference

template<typename Subsys, typename Config = TCSConfig> class TCSApp< Subsys, Config >

Public Member Functions

- **TCSApp** (int argc, char **argv, const string &appName, bool fpeExcept=true)
- int run (bool addAllRoutes=false, bool addAllEventRoutes=false)

  Starts the HET application, performs basic configuration, and starts the application's message handlers. Waits for an incoming message.

Public Attributes

- int **argc_**
- char ** **argv_**
- string **appName_**
- Config * **config_**
- Subsys * **subSystem_**
- bool **fpe_**

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_app.h

9.896 TCSBaseClass Class Reference

9.896.1 Detailed Description

A base class for many PAS objects

`#include <tcs_baseclass.h>`

Public Types

- enum **State_t** { **UNINITIALIZED**, **INITIALIZING**, **ONLINE**, **STOPPED** }

Public Member Functions

- **TCSBaseClass** (TCSConfig *pConfig)
- void **SetName** (const string &name)
- const char * **GetName** (void)
- void **SetDebug** (bool on)
- void **SetState** (State_t state)
- State_t **GetState** (void)

Protected Attributes

- std::string **m_Name**
- State_t **m_State**
- bool **m_Debug**
- TCSConfig * **m_pConfig**

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_baseclass.h

9.897 TCSBaseException Class Reference

Public Member Functions

- **TCSBaseException** (tcsexcept::tcs_exception_t t, const std::string &w, const char *file, int line, const char *func)
- virtual const char * **what** () const throw ()
- std::string **file** ()
- std::string **func** ()
- tcsexcept::tcs_exception_t **type** ()
- int **line** ()
- virtual void **print** (std::ostream &o=std::cerr) const

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_exception.h

9.898 TCSClient Class Reference

Classes

- struct TelescopeData

Public Member Functions

- **TCSClient** (const std::string &route, const std::string &name)
- void **write** (FitsOutWrapper &f)
- void **update_state** ()
- void **reinitialize** ()
- **TCSClient** (const std::string &route, const std::string &name)
- void **get_telescope_data** (std::string cameraTypeName, pas::Camera_t cameraType, TelescopeData &data)
- bool **select_camera_filter** (std::string cameraTypeName, int filter)

The documentation for this class was generated from the following files:

- camra/tcs_client.h
- pas/include/tcs_client.h
- camra/tcs_client.cpp
- pas/src/tcs_client.cpp

9.899 tcsCmd Class Reference

Public Member Functions

- **tcsCmd** (const string &appName, cmdConfig *config)
- virtual int **start** (void)

Protected Attributes

- cmdConfig * **m_pConfig**
- Sender * **m_pSender**

The documentation for this class was generated from the following file:

- tcsconsole/tcscmd.cpp

9.900 TCSConfig Class Reference

Configuration class for HET applcations.

9.900.1 Detailed Description

Configuration class that provides a set of default command line and configuration file options and standardizes the behavior of command line override. Application developers inherit this class with application configuration classes and implement the TCSConfig::set_options() and TCSConfig::print_summary() methods to extend the options supported by the application.

`#include <tcs_config.h>`

Classes

- struct mapEntry

Public Types

- enum **lookupType_t** { **lookupTypeString**, **lookupTypeInteger** }
- typedef std::map< string, mapEntry * > **lookupMap_t**

Public Member Functions

- TCSConfig (bool named, const char *who)

  Base constructor.

- virtual void set_options (OptionParser &optionParser)=0

  Adds options to default set.

- virtual void print_summary (void)=0

  Called prior to displaying options.

- void configure (const string &named_route, const string &config_file, const string &config_root="", bool addAllRoutes=false, bool addAllEventRoutes=false)

  Configures an instance of the sub-class.

- void configure (int argc, char **argv, bool addAllRoutes=false, bool addAllEventRoutes=false)

  Configures an instance of the sub-class.

- void **AddRuntimeConstructedLookupEntry** (lookupType_t type, void *var, const string &key)
- string **lookupRoute** (const string &route)
- string **lookupEventRoute** (const string &route)
- string **lookupString** (const string &key)
- int **lookupInt** (const string &key)

Public Attributes

- lookupMap_t **runtimeConstructedlookupMap**
- bool version

  Flag indicating that the application version has been requested.

- bool help

  Flag indicating that the application usage has been requested.

- bool debug

  Flag indicating that the application debugging has been requested.

- string config_root

  The root path of a configuration file.

- char hostname [512]

  The hostname of the computer the application is running on.

- string name

  The name of the configuration object.

- const char * program

  The file name of the application.

- list< string > extra_arguments

  The extra (non keyword) arguments (filenames).

- int log_levels

  Bit field of log levels.

- string **log_levels_str**
- string **named_route**
- string **named_event_route**
- string **apc_route**
- string **apc_event_route**
- string **tcs_route**
- string **tcs_event_route**
- string **pas_route**
- string **pas_event_route**
- string **tracker_route**
- string **tracker_event_route**
- string **legacy_route**
- string **legacy_event_route**
- string **pfip_route**
- string **pfip_event_route**
- string **gui_route**
- string **gui_event_route**
- string **camra_route**
- string **camra_event_route**
- string **lrs2_route**

- string **lrs2_event_route**
- string **virus_route**
- string **virus_event_route**
- string **plotter_route**
- string **plotter_event_route**
- string **monitor_route**
- string **monitor_event_route**
- string **log_relay_route**
- string **log_relay_event_route**
- string **event_monitor_route**
- string **event_monitor_event_route**
- string **epics_relay_route**
- string **epics_relay_event_route**
- string **pfip_epics_relay_route**
- string **pfip_epics_relay_event_route**
- string **thermocube_spare_epics_relay_route**
- string **thermocube_spare_epics_relay_event_route**
- string **thermocube_side1_epics_relay_route**
- string **thermocube_side1_epics_relay_event_route**
- string **thermocube_side2_epics_relay_route**
- string **thermocube_side2_epics_relay_event_route**
- string **ocd_route**
- string **ocd_event_route**
- string **igloo_sensors_route**
- string **igloo_sensors_event_route**
- string **sample_route**
- string **sample_event_route**
- string **dimmpoller_route**
- string **dimmpoller_event_route**
- string **pfipmaxon_route**
- string **pfipmaxon_event_route**
- string **ser7_path**
- string **guider_markers_path**
- string **structure_model_path**
- string **pfip_lamps_database_path**

9.900.2 Constructor & Destructor Documentation

9.900.2.1 TCSConfig::TCSConfig (bool *named*, const char * *who*)

This should be called with the `_PROGRAM` macro set at compile time of the application. This allows the compile time version information to propagate to the statically linked library.

9.900.3 Member Function Documentation

9.900.3.1 void TCSConfig::configure (int *argc*, char ** *argv*, bool *addAllRoutes* = `false`, bool *addAllEventRoutes* = `false`)

Applies the configuration parsed from the command line arguments, in addition to any configuration referenced in the arguments, to the TCSConfig sub-class instance. Any configuration file options are applied first and then overriden by the corresponding command line options (if present).

9.900.3.2 void TCSConfig::configure (const string & *named_route*, const string & *config_file*, const string & *config_root* = "", bool *addAllRoutes* = false, bool *addAllEventRoutes* = false)

Applies the configuration parsed from the given `config_file` to the TCSConfig sub-class instance. Applications may use this method to parse a generic configuration file that is not coupled to the command line or to build hierarchical configuration structures from config and "sub-config" files.

If the config_root is given, then the config_file is searched for at the given root.

9.900.3.3 virtual void TCSConfig::print_summary (void) [pure virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implemented in apcCmdConfig, GroupConfig, apcConfig, CAMRAConfig, ControllerConfig, SpectrographConfig, WriteOneWireConfig, EPICSRelayConfig, guiConfig, SubSystemConfig, IglooSensorsConfig, legacyConfig, pasConfig, PFIPConfig, sampleConfig, pointingConfig, ProxyConfig, TCSServerConfig, cmdConfig, consoleConfig, eventsConfig, namedConfig, TrackerConfig, and guiConfig.

9.900.3.4 virtual void TCSConfig::set_options (OptionParser & *optionParser*) [pure virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implemented in apcCmdConfig, GroupConfig, pduConfig, apcConfig, CAMRAConfig, ControllerConfig, SpectrographConfig, WriteOneWireConfig, EPICSRelayConfig, guiConfig, SubSystemConfig, IglooSensorsConfig, DomeConfig, ShutterConfig, StructureConfig, WeatherConfig, legacyConfig, LightSourceConfig, CameraConfig, CleanerConfig, ProcessorConfig, DeviceConfig, PipelineConfig, pasConfig, PFIPConfig, sampleConfig, pointingConfig, ProxyConfig, TCSServerConfig, cmdConfig, consoleConfig, eventsConfig, namedConfig, TrackerConfig, and guiConfig.

9.900.4 Member Data Documentation

9.900.4.1 string TCSConfig::config_root

If the config_root is not explicitly given in a call to configure(), then it is derived from the directory name of the configuration file given.

9.900.4.2 bool TCSConfig::debug

Corresponds to "debug" in the configuration file and '-d' on the command line, defaults to FALSE.

9.900.4.3 bool TCSConfig::help

Corresponds to "help" in the configuration file and '-h' on the command line, defaults to FALSE.

9.900.4.4 char TCSConfig::hostname[512]

Set when `TCSConfig::configure()` is called.

9.900.4.5 int TCSConfig::log_levels

Corresponds to a comma separated list of log levels that will be set upon Logger configuration.

See Logger for documentation of the supported log level strings.

9.900.4.6 string TCSConfig::name

Can be used for "name" in the configuration file.

9.900.4.7 const char* TCSConfig::program

Set when `TCSConfig::configure()` is called.

9.900.4.8 bool TCSConfig::version

Corresponds to "version" in the configuration file and '-v' on the command line, defaults to FALSE.

The documentation for this class was generated from the following files:

- het_lib/common/include/tcs_config.h
- het_lib/common/src/tcs_config.cpp

9.901 TCSConfigException Class Reference

Public Member Functions

- **TCSConfigException** (tcsexcept::tcs_exception_t __config, const std::string &w, const char *file, int line, const char *func)

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_exception.h

9.902 tcsConsole Class Reference

Public Member Functions

- **tcsConsole** (const string &appName, consoleConfig *config)
- virtual int **start** (void)

Protected Attributes

- consoleConfig * **m_pConfig**
- Sender * **m_pSender**

The documentation for this class was generated from the following file:

- tcsconsole/tcsconsole.cpp

9.903 pasImagePipelineSubSystem::TCSData Struct Reference

Classes

- struct fiducialObject

Public Attributes

- tthread::mutex **accessMutex**
- struct {
  int **xMin**
  int **yMin**
  int **xMax**
  int **yMax**
  } **analysisRegion**

- struct pasImagePipelineSubSystem::TCSData::fiducialObject **fiducialObject**
- struct {
  bool **enabled**
  bool **active**
  } **metrologyLoopState**

The documentation for this struct was generated from the following file:

- pas/include/pasImagePipelineSubSystem.h

9.904 tcsdb::tcsdb Class Reference

Public Member Functions

- def **__init__**
- def **keys**
- def **events**
- def **start**
- def **stop**
- def **__call__**
- def **__del__**

Public Attributes

- **db_**
- **cursor_**
- **events_**
- **event_ids_**
- **event_keys_**
- **start_**
- **stop_**

The documentation for this class was generated from the following file:

- scripting/pytcs/build/lib.linux-x86_64-2.7/tcsdb.py

9.905 pytcs::tcsdb::tcsdb Class Reference

Public Member Functions

- def **__init__**
- def **keys**
- def **events**
- def **start**
- def **stop**
- def **__call__**
- def **__del__**

Public Attributes

- **db_**
- **cursor_**
- **events_**
- **event_ids_**
- **event_keys_**
- **start_**
- **stop_**

The documentation for this class was generated from the following file:

- scripting/pytcs/tcsdb.py

9.906 TCSDriver Class Reference

9.906.1 Detailed Description

TCS "driver" class that encapsulate all interfaces to the TCS hardware from the perspective of the Telescope Control System (TCS).

`#include <tcs_driver.h>`

Public Member Functions

- **TCSDriver** (const string &appName, TCSServerConfig *config)
- int **start** ()
- void **stop** ()
- Message * **toggle_gui_token** (string &token)
- Message * **load_trajectory** (Message *message, bool debug=0)
- void **offset_trajectory** (double dra, double ddec, bool adjust_probes, bool correction, fptime &when)
- void **correct_for_structure** ()
- vec2_t get_radec_offset_register (const pointing_coordinate &p)
- void reset_radec_offset_register (void)
- itf_coordinate **get_itf_offset** (void)
- void **set_itf_offset** (const itf_coordinate &offset, bool have_lock=0)
- void **reset_itf_offset** ()
- vec2_t **get_sky_offset** (void)
- void **set_sky_offset** (const vec2_t &offset, bool adjust_probes)
- void **reset_sky_offset** (bool adjust_probes)
- double **get_focus_offset** (void)
- void **set_focus_offset** (const double &offset)
- void **reset_focus_offset** ()
- double **get_rho_offset** (void)
- void **set_rho_offset** (const double &offset)
- void **reset_rho_offset** ()
- vec2_t **get_tiptilt_offset** (void)
- void **set_tiptilt_offset** (const vec2_t &offset)
- void **reset_tiptilt_offset** ()
- vec2_t **get_sky_correction** (void)
- void **set_sky_correction** (const vec2_t &correction)
- void **reset_sky_correction** (void)
- vec2_t **get_empirical_correction** (void)
- void **set_empirical_correction** (const vec2_t &correction)
- void **reset_empirical_correction** (void)
- void **set_focus** (const double &w)
- void go_next (GoNextFlags &flags)
- void cancel_trajectory (bool do_throw=1)
- void clear_trajectories (void)
- Message * trajectory_info (traj_ptr_t which_trajectory)
- bool **get_pointing** (pointing_coordinate &p, bool do_exception=0)
- void **get_pointing_payload** (Payload &p, bool do_exception=0)
- pointing_coordinate **pointing** (void)

- Receiver & **receiver** (void)
- TrackerClient & **tracker** ()
- PFIPClient & **pfip** ()
- DomeClient & **dome** ()
- StructureClient & **structure** ()
- WeatherClient & **weather** ()
- Trajectory & trajectory (bool force_next)
- Metrology & **metrology** ()
- itf_coordinate **adjust_for_offsets** (const itf_coordinate &original)
- bool **offsetting** ()
- void **setup** (bool state)
- void **wait_for_setup** (double timeout=-1)
- void **set_offsetting** ()
- void **unset_offsetting** (bool update_last_offset_time=1)
- void **set_metadata** (const string &key, const string &value, const string &type)
- void **get_metadata** (Payload &p)
- void **reset_metadata** (void)
- fptime **get_last_offset_time** (void)
- void **enable_rho** (void)
- void **disable_rho** (void)

9.906.2 Member Function Documentation

9.906.2.1 void TCSDriver::cancel_trajectory (bool *do_throw* = 1)

Cancel the current running or queued trajectory.

9.906.2.2 void TCSDriver::clear_trajectories (void)

Clear all loaded trajectories.

9.906.2.3 vec2_t TCSDriver::get_radec_offset_register (const pointing_coordinate & *p*)

Measures cumulative trajectory offsets relative to register. Upon trajectory start, the register is intitialized with the commanded apparent coordinates

9.906.2.4 void TCSDriver::go_next (GoNextFlags & *flags*)

Activate the next queued trajectory.

9.906.2.5 void TCSDriver::reset_radec_offset_register (void)

Resets the register to the current apparent cooridiates. No offset to the trajectory is applied

9.906.2.6 Trajectory & TCSDriver::trajectory (bool *force_next*)

If there is a current trajectory, returns that. Otherwise the next trajectory made current and that one is returned. If force_next is true, the next trajectory is returned without making it current.

9.906.2.7 Message * TCSDriver::trajectory_info (traj_ptr_t *which_trajectory*)

Returns a Message with parameters initialized to the current state of the given Trajectory.

XXX where do we document the contents of a Message returned?

The documentation for this class was generated from the following files:

- tcs/tcs_driver.h
- tcs/tcs_driver.cpp

9.907 TCSEvent::TCSEvent Class Reference

Public Member Functions

- def **__init__**
- def **createEventSource**
- def **newPayload**
- def **setParm**
- def **sendEvent**
- def **event**
- def **deleteEventSource**
- def **__exit__**

The documentation for this class was generated from the following file:

- playground/TCSEvent/TCSEvent.py

9.908 tcs_event::TCSEvent Class Reference

Public Member Functions

- def __init__
- def subscribe
- def connect
- def next

9.908.1 Member Function Documentation

9.908.1.1 def tcs_event::TCSEvent::__init__ (*self*, *urls* = None, *topics* = None)

```
TCS event class that connects to the given list of urls and listens
for events. See next().

WARNING: This object currently does not support concurrency.
```

9.908.1.2 def tcs_event::TCSEvent::connect (*self*, *urls*)

```
Connect to each of the urls in the given list.  Note that this does
not affect any prior connections.  However, receipt order is still at
the whim of the producers.
```

9.908.1.3 def tcs_event::TCSEvent::next (*self*, *timeout_ms* = -1, *topics* = None)

```
Returns a tuple containing the next event topic and dictionary,
matching any requested topics, received.  Blocks indefinitely.

XXX add timeout?

Example:

import tcsevent
events = tcsevent.tcsevent(["tcp://127.0.0.1:30301"],
["tcs.tracker.position",
"tcs.root.ra_dec"] )
while True:
[h,e] = events.next()
if e is not None:
break
# handle e
```

9.908.1.4 def tcs_event::TCSEvent::subscribe (*self*, *topics*)

```
Subscribe to each of the topics in the given list. Note that this
does not affect any prior subscriptions.
```

The documentation for this class was generated from the following file:

- scripting/pytcs/build/lib.linux-x86_64-2.7/tcs_event.py

9.909 pytcs::tcs_event::TCSEvent Class Reference

Public Member Functions

- def __init__
- def subscribe
- def connect
- def next

9.909.1 Member Function Documentation

9.909.1.1 def pytcs::tcs_event::TCSEvent::__init__ (*self*, *urls* = `None`, *topics* = `None`)

```
TCS event class that connects to the given list of urls and listens
for events. See next().

WARNING: This object currently does not support concurrency.
```

9.909.1.2 def pytcs::tcs_event::TCSEvent::connect (*self*, *urls*)

```
Connect to each of the urls in the given list.  Note that this does
not affect any prior connections.  However, receipt order is still at
the whim of the producers.
```

9.909.1.3 def pytcs::tcs_event::TCSEvent::next (*self*, *timeout_ms* = `-1`, *topics* = `None`)

```
Returns a tuple containing the next event topic and dictionary,
matching any requested topics, received.  Blocks indefinitely.

XXX add timeout?

Example:

import tcsevent
events = tcsevent.tcsevent(["tcp://127.0.0.1:30301"],
["tcs.tracker.position",
"tcs.root.ra_dec"] )
while True:
[h,e] = events.next()
if e is not None:
break
# handle e
```

9.909.1.4 def pytcs::tcs_event::TCSEvent::subscribe (*self*, *topics*)

```
Subscribe to each of the topics in the given list. Note that this
does not affect any prior subscriptions.
```

The documentation for this class was generated from the following file:

- scripting/pytcs/tcs_event.py

9.910 tcsEvents Class Reference

Public Member Functions

- **tcsEvents** (const string &appName, eventsConfig *config)
- virtual int **start** (void)

Protected Attributes

- eventsConfig * **m_pConfig**
- EventSourceRoot * **m_pEventSourceRoot**
- EventSource * **m_pEventSource**

The documentation for this class was generated from the following file:

- tcsconsole/tcsevents.cpp

9.911 TCSException Class Reference

Public Member Functions

- **TCSException** (tcsexcept::tcs_exception_t __tcs, const std::string &w, const char ∗file, int line, const char ∗func)

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_exception.h

9.912 TcsException Class Reference

Public Member Functions

- **TcsException** (const char ∗format,...)
- virtual const char ∗ **what** () const throw ()

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_subsystem.h

9.913 tcsGui Class Reference

Classes

- class soundObject

Public Types

- typedef std::map< string, fptime > **heartbeat_t**
- typedef std::map< string, bool > **subSystemRunning_t**
- typedef std::map< string, bool > **subSystemForceHeartbeat_t**

Public Member Functions

- **tcsGui** (const string &appName, guiConfig *pConfig)
- virtual int **start** (void)
- void **saveSettings** (void)
- guiConfig * **getConfig** (void)
- QApplication * **getApp** (void)
- msgid_t **getLocalMsgId** (void)
- pas::Camera_t **getCameraTypeFromPipelineName** (string pipelineName)
- pas::ProcessingMode_t **getProcessingModeFromPipelineName** (string pipelineName)
- void **showSplashScreenMessage** (QString msg, int alignment=Qt::AlignTop|Qt::AlignHCenter, const QColor=Qt::white)

Public Attributes

- QString **m_loginId**
- QString **m_SettingsFilename**
- QString **m_StyleSheetFilename**
- QSettings * **m_pSettings**
- guiPreferences **m_preferences**
- StatusBlock< TrackerStatus > * **m_pTrackerStatus**
- StatusBlock< TrackerPosition > * **m_pTrackerPosition**
- StatusBlock< DomeStatus > * **m_pDomeStatus**
- StatusBlock< ShutterStatus > * **m_pShutterStatus**
- StatusBlock< StructureStatus > * **m_pStructureStatus**
- StatusBlock< WeatherData > * **m_pWeatherData**
- StatusBlock< DMIData > * **m_pDMIData**
- StatusBlock< SAMSData > * **m_pSAMSData**
- StatusBlock< ProbeStatus > * **m_pProbeStatus** [pfip::NUMBER_OF_PROBES]
- StatusBlock< AcqCamStatus > * **m_pAcqCamStatus**
- StatusBlock< AcqShutterStatus > * **m_pAcqShutterStatus**
- StatusBlock< AcqENWAStatus > * **m_pAcqENWAStatus**
- StatusBlock< AcqCWFSStatus > * **m_pAcqCWFSStatus**
- StatusBlock< AcqMiscStatus > * **m_pAcqMiscStatus**
- StatusBlock< FCUStatus > * **m_pFCUStatus**
- StatusBlock< PFIPMonitorStatus > * **m_pPFIPMonitorStatus**
- StatusBlock< VirusMonitorStatus > * **m_pVirusMonitor1Status**

- StatusBlock< VirusMonitorStatus > * **m_pVirusMonitor2Status**
- StatusBlock< APCData > * **m_pAPCData**
- StatusBlock< OCDStatus > * **m_pOCDStatus**
- StatusBlock< ThermoCubeStatus > * **m_pThermoCubeSide1Status**
- StatusBlock< ThermoCubeStatus > * **m_pThermoCubeSide2Status**
- StatusBlock< IglooData > * **m_pIglooData**
- tthread::mutex **m_heartbeatAccessMutex**
- heartbeat_t **m_heartbeat**
- subSystemRunning_t **m_subSystemRunning**
- subSystemForceHeartbeat_t **m_subSystemForceHeartbeat**
- QTimer * **m_pSecondTimer**
- tcsGuiThreadWorker * **m_pMonitorHeartbeatsThreadWorker**
- tcsGuiThreadWorker * **m_pCommandReceiverThreadWorker**
- tcsGuiThreadWorker * **m_pSoundServerThreadWorker**
- tcsGuiThreadWorker * **m_pForceHeartbeatThreadWorker**
- tcsGuiThreadWorker * **m_pSoundThreadWorker**
- tcsGuiThreadWorker * **m_pRelaySignalsThreadWorker**
- tcsGuiThreadWorker * **m_pReceiveStatusEventsThreadWorker**
- tcsGuiThreadWorker * **m_pReceiveHeartbeatEventsThreadWorker**
- tcsGuiThreadWorker * **m_pSendTcsAsyncCommandsThreadWorker**
- tcsGuiThreadWorker * **m_pReceiveTcsAsyncRepliesThreadWorker**
- tcsGuiThreadWorker * **m_pSendTrackerAsyncCommandsThreadWorker**
- tcsGuiThreadWorker * **m_pReceiveTrackerAsyncRepliesThreadWorker**
- tcsGuiThreadWorker * **m_pSendLegacyAsyncCommandsThreadWorker**
- tcsGuiThreadWorker * **m_pReceiveLegacyAsyncRepliesThreadWorker**
- tcsGuiThreadWorker * **m_pSendPasAsyncCommandsThreadWorker**
- tcsGuiThreadWorker * **m_pReceivePasAsyncRepliesThreadWorker**
- tcsGuiThreadWorker * **m_pSendPfipAsyncCommandsThreadWorker**
- tcsGuiThreadWorker * **m_pReceivePfipAsyncRepliesThreadWorker**
- tcsGuiThreadWorker * **m_pSendApcAsyncCommandsThreadWorker**
- tcsGuiThreadWorker * **m_pReceiveApcAsyncRepliesThreadWorker**
- tcsGuiThreadWorker * **m_pSendVirusAsyncCommandsThreadWorker**
- tcsGuiThreadWorker * **m_pReceiveVirusAsyncRepliesThreadWorker**
- tcsGuiThreadWorker * **m_pSendLrs2AsyncCommandsThreadWorker**
- tcsGuiThreadWorker * **m_pReceiveLrs2AsyncRepliesThreadWorker**
- tcsGuiThreadWorker * **m_pSendIglooAsyncCommandsThreadWorker**
- tcsGuiThreadWorker * **m_pReceiveIglooAsyncRepliesThreadWorker**

Protected Member Functions

- void **login** (void)
- void **loadSettings** (void)
- void **MonitorHeartbeatsThreadEntryPoint** (void)
- void **CommandReceiverThreadEntryPoint** (void)
- void **SoundServerThreadEntryPoint** (void)
- void **ForceHeartbeatThreadEntryPoint** (void)
- void **SoundThreadEntryPoint** (void)
- void **RelaySignalsThreadEntryPoint** (void)
- void **ReceiveStatusEventsThreadEntryPoint** (void)
- void **ReceiveHeartbeatEventsThreadEntryPoint** (void)

- void **SendTcsAsyncCommandsThreadEntryPoint** (void)
- void **ReceiveTcsAsyncRepliesThreadEntryPoint** (void)
- void **SendTrackerAsyncCommandsThreadEntryPoint** (void)
- void **ReceiveTrackerAsyncRepliesThreadEntryPoint** (void)
- void **SendLegacyAsyncCommandsThreadEntryPoint** (void)
- void **ReceiveLegacyAsyncRepliesThreadEntryPoint** (void)
- void **SendPasAsyncCommandsThreadEntryPoint** (void)
- void **ReceivePasAsyncRepliesThreadEntryPoint** (void)
- void **SendPfipAsyncCommandsThreadEntryPoint** (void)
- void **ReceivePfipAsyncRepliesThreadEntryPoint** (void)
- void **SendApcAsyncCommandsThreadEntryPoint** (void)
- void **ReceiveApcAsyncRepliesThreadEntryPoint** (void)
- void **SendVirusAsyncCommandsThreadEntryPoint** (void)
- void **ReceiveVirusAsyncRepliesThreadEntryPoint** (void)
- void **SendLrs2AsyncCommandsThreadEntryPoint** (void)
- void **ReceiveLrs2AsyncRepliesThreadEntryPoint** (void)
- void **SendIglooAsyncCommandsThreadEntryPoint** (void)
- void **ReceiveIglooAsyncRepliesThreadEntryPoint** (void)

Protected Attributes

- tthread::thread * **m_pMonitorHeartbeatsThread**
- volatile bool **m_bMonitorHeartbeatsThreadRunning**
- tthread::mutex **m_MonitorHeartbeatsThreadAccessMutex**
- guiThreadState_t **m_MonitorHeartbeatsThreadState**
- string **m_MonitorHeartbeatsThreadStartupError**
- tthread::thread * **m_pCommandReceiverThread**
- volatile bool **m_bCommandReceiverThreadRunning**
- tthread::mutex **m_CommandReceiverThreadAccessMutex**
- guiThreadState_t **m_CommandReceiverThreadState**
- string **m_CommandReceiverThreadStartupError**
- tthread::thread * **m_pSoundServerThread**
- volatile bool **m_bSoundServerThreadRunning**
- tthread::mutex **m_SoundServerThreadAccessMutex**
- guiThreadState_t **m_SoundServerThreadState**
- string **m_SoundServerThreadStartupError**
- tthread::thread * **m_pForceHeartbeatThread**
- volatile bool **m_bForceHeartbeatThreadRunning**
- tthread::mutex **m_ForceHeartbeatThreadAccessMutex**
- guiThreadState_t **m_ForceHeartbeatThreadState**
- string **m_ForceHeartbeatThreadStartupError**
- QEventLoop * **m_pForceHeartbeatEventLoop**
- tthread::thread * **m_pSoundThread**
- volatile bool **m_bSoundThreadRunning**
- tthread::mutex **m_SoundThreadAccessMutex**
- guiThreadState_t **m_SoundThreadState**
- string **m_SoundThreadStartupError**
- QEventLoop * **m_pSoundEventLoop**
- tthread::thread * **m_pRelaySignalsThread**
- volatile bool **m_bRelaySignalsThreadRunning**

- tthread::mutex **m_RelaySignalsThreadAccessMutex**
- guiThreadState_t **m_RelaySignalsThreadState**
- string **m_RelaySignalsThreadStartupError**
- QEventLoop * **m_pRelaySignalsEventLoop**
- tthread::thread * **m_pReceiveStatusEventsThread**
- volatile bool **m_bReceiveStatusEventsThreadRunning**
- tthread::mutex **m_ReceiveStatusEventsThreadAccessMutex**
- guiThreadState_t **m_ReceiveStatusEventsThreadState**
- string **m_ReceiveStatusEventsThreadStartupError**
- EventSink * **m_pReceiveStatusEventsSink**
- tthread::thread * **m_pReceiveHeartbeatEventsThread**
- volatile bool **m_bReceiveHeartbeatEventsThreadRunning**
- tthread::mutex **m_ReceiveHeartbeatEventsThreadAccessMutex**
- guiThreadState_t **m_ReceiveHeartbeatEventsThreadState**
- string **m_ReceiveHeartbeatEventsThreadStartupError**
- EventSink * **m_pReceiveHeartbeatEventsSink**
- tthread::thread * **m_pSendTcsAsyncCommandsThread**
- volatile bool **m_bSendTcsAsyncCommandsThreadRunning**
- tthread::mutex **m_SendTcsAsyncCommandsThreadAccessMutex**
- guiThreadState_t **m_SendTcsAsyncCommandsThreadState**
- string **m_SendTcsAsyncCommandsThreadStartupError**
- QEventLoop * **m_pSendTcsAsyncCommandsEventLoop**
- tthread::thread * **m_pReceiveTcsAsyncRepliesThread**
- volatile bool **m_bReceiveTcsAsyncRepliesThreadRunning**
- tthread::mutex **m_ReceiveTcsAsyncRepliesThreadAccessMutex**
- guiThreadState_t **m_ReceiveTcsAsyncRepliesThreadState**
- string **m_ReceiveTcsAsyncRepliesThreadStartupError**
- Sender * **m_pTcsAsyncCommandSender**
- tthread::thread * **m_pSendTrackerAsyncCommandsThread**
- volatile bool **m_bSendTrackerAsyncCommandsThreadRunning**
- tthread::mutex **m_SendTrackerAsyncCommandsThreadAccessMutex**
- guiThreadState_t **m_SendTrackerAsyncCommandsThreadState**
- string **m_SendTrackerAsyncCommandsThreadStartupError**
- QEventLoop * **m_pSendTrackerAsyncCommandsEventLoop**
- tthread::thread * **m_pReceiveTrackerAsyncRepliesThread**
- volatile bool **m_bReceiveTrackerAsyncRepliesThreadRunning**
- tthread::mutex **m_ReceiveTrackerAsyncRepliesThreadAccessMutex**
- guiThreadState_t **m_ReceiveTrackerAsyncRepliesThreadState**
- string **m_ReceiveTrackerAsyncRepliesThreadStartupError**
- Sender * **m_pTrackerAsyncCommandSender**
- tthread::thread * **m_pSendLegacyAsyncCommandsThread**
- volatile bool **m_bSendLegacyAsyncCommandsThreadRunning**
- tthread::mutex **m_SendLegacyAsyncCommandsThreadAccessMutex**
- guiThreadState_t **m_SendLegacyAsyncCommandsThreadState**
- string **m_SendLegacyAsyncCommandsThreadStartupError**
- QEventLoop * **m_pSendLegacyAsyncCommandsEventLoop**
- tthread::thread * **m_pReceiveLegacyAsyncRepliesThread**
- volatile bool **m_bReceiveLegacyAsyncRepliesThreadRunning**
- tthread::mutex **m_ReceiveLegacyAsyncRepliesThreadAccessMutex**
- guiThreadState_t **m_ReceiveLegacyAsyncRepliesThreadState**

- string **m_ReceiveLegacyAsyncRepliesThreadStartupError**
- Sender * **m_pLegacyAsyncCommandSender**
- tthread::thread * **m_pSendPasAsyncCommandsThread**
- volatile bool **m_bSendPasAsyncCommandsThreadRunning**
- tthread::mutex **m_SendPasAsyncCommandsThreadAccessMutex**
- guiThreadState_t **m_SendPasAsyncCommandsThreadState**
- string **m_SendPasAsyncCommandsThreadStartupError**
- QEventLoop * **m_pSendPasAsyncCommandsEventLoop**
- tthread::thread * **m_pReceivePasAsyncRepliesThread**
- volatile bool **m_bReceivePasAsyncRepliesThreadRunning**
- tthread::mutex **m_ReceivePasAsyncRepliesThreadAccessMutex**
- guiThreadState_t **m_ReceivePasAsyncRepliesThreadState**
- string **m_ReceivePasAsyncRepliesThreadStartupError**
- Sender * **m_pPasAsyncCommandSender**
- tthread::thread * **m_pSendPfipAsyncCommandsThread**
- volatile bool **m_bSendPfipAsyncCommandsThreadRunning**
- tthread::mutex **m_SendPfipAsyncCommandsThreadAccessMutex**
- guiThreadState_t **m_SendPfipAsyncCommandsThreadState**
- string **m_SendPfipAsyncCommandsThreadStartupError**
- QEventLoop * **m_pSendPfipAsyncCommandsEventLoop**
- tthread::thread * **m_pReceivePfipAsyncRepliesThread**
- volatile bool **m_bReceivePfipAsyncRepliesThreadRunning**
- tthread::mutex **m_ReceivePfipAsyncRepliesThreadAccessMutex**
- guiThreadState_t **m_ReceivePfipAsyncRepliesThreadState**
- string **m_ReceivePfipAsyncRepliesThreadStartupError**
- Sender * **m_pPfipAsyncCommandSender**
- tthread::thread * **m_pSendApcAsyncCommandsThread**
- volatile bool **m_bSendApcAsyncCommandsThreadRunning**
- tthread::mutex **m_SendApcAsyncCommandsThreadAccessMutex**
- guiThreadState_t **m_SendApcAsyncCommandsThreadState**
- string **m_SendApcAsyncCommandsThreadStartupError**
- QEventLoop * **m_pSendApcAsyncCommandsEventLoop**
- tthread::thread * **m_pReceiveApcAsyncRepliesThread**
- volatile bool **m_bReceiveApcAsyncRepliesThreadRunning**
- tthread::mutex **m_ReceiveApcAsyncRepliesThreadAccessMutex**
- guiThreadState_t **m_ReceiveApcAsyncRepliesThreadState**
- string **m_ReceiveApcAsyncRepliesThreadStartupError**
- Sender * **m_pApcAsyncCommandSender**
- tthread::thread * **m_pSendVirusAsyncCommandsThread**
- volatile bool **m_bSendVirusAsyncCommandsThreadRunning**
- tthread::mutex **m_SendVirusAsyncCommandsThreadAccessMutex**
- guiThreadState_t **m_SendVirusAsyncCommandsThreadState**
- string **m_SendVirusAsyncCommandsThreadStartupError**
- QEventLoop * **m_pSendVirusAsyncCommandsEventLoop**
- tthread::thread * **m_pReceiveVirusAsyncRepliesThread**
- volatile bool **m_bReceiveVirusAsyncRepliesThreadRunning**
- tthread::mutex **m_ReceiveVirusAsyncRepliesThreadAccessMutex**
- guiThreadState_t **m_ReceiveVirusAsyncRepliesThreadState**
- string **m_ReceiveVirusAsyncRepliesThreadStartupError**
- Sender * **m_pVirusAsyncCommandSender**

- tthread::thread * **m_pSendLrs2AsyncCommandsThread**
- volatile bool **m_bSendLrs2AsyncCommandsThreadRunning**
- tthread::mutex **m_SendLrs2AsyncCommandsThreadAccessMutex**
- guiThreadState_t **m_SendLrs2AsyncCommandsThreadState**
- string **m_SendLrs2AsyncCommandsThreadStartupError**
- QEventLoop * **m_pSendLrs2AsyncCommandsEventLoop**
- tthread::thread * **m_pReceiveLrs2AsyncRepliesThread**
- volatile bool **m_bReceiveLrs2AsyncRepliesThreadRunning**
- tthread::mutex **m_ReceiveLrs2AsyncRepliesThreadAccessMutex**
- guiThreadState_t **m_ReceiveLrs2AsyncRepliesThreadState**
- string **m_ReceiveLrs2AsyncRepliesThreadStartupError**
- Sender * **m_pLrs2AsyncCommandSender**
- tthread::thread * **m_pSendIglooAsyncCommandsThread**
- volatile bool **m_bSendIglooAsyncCommandsThreadRunning**
- tthread::mutex **m_SendIglooAsyncCommandsThreadAccessMutex**
- guiThreadState_t **m_SendIglooAsyncCommandsThreadState**
- string **m_SendIglooAsyncCommandsThreadStartupError**
- QEventLoop * **m_pSendIglooAsyncCommandsEventLoop**
- tthread::thread * **m_pReceiveIglooAsyncRepliesThread**
- volatile bool **m_bReceiveIglooAsyncRepliesThreadRunning**
- tthread::mutex **m_ReceiveIglooAsyncRepliesThreadAccessMutex**
- guiThreadState_t **m_ReceiveIglooAsyncRepliesThreadState**
- string **m_ReceiveIglooAsyncRepliesThreadStartupError**
- Sender * **m_pIglooAsyncCommandSender**
- Receiver * **m_pCommandReceiver**
- QApplication * **m_pQtApp**
- QSplashScreen * **m_pSplashScreen**
- guiConfig * **m_pConfig**
- int **m_localMsgId**
- tthread::mutex **m_localMsgIdMutex**
- std::list< soundObject > **m_SoundQueue**
- tthread::mutex **m_SoundQueueMutex**
- tthread::condition_variable **m_SoundQueueDataAvailable**

Friends

- class tcsGuiThreadWorker

The documentation for this class was generated from the following files:

- gui/include/tcsGui.h
- gui/src/tcsGui.cpp

9.914 tcsGuiThreadWorker Class Reference

Public Slots

- void **InitTracker** (void)
- void **AbortTracker** (void)
- void **ClearTrackerErrors** (int)
- void **MoveTrackerITF** (std::string, std::string, double, double, double, double, double, double)
- void **MoveTrackerRTF** (std::string, std::string, double, double, double, double, double, double)
- void **MoveTrackerTSF** (std::string, std::string, double, double, double, double, double, double, double, double, double)
- void **MoveTrackerAng** (std::string, std::string, double, double)
- void **LoadRaDecTrajectory** (double, double, double, std::string, double, double)
- void **LoadXYTrajectory** (double, double, double)
- void **LoadFileTrajectory** (std::string, double, double)
- void **LoadSatelliteTrajectory** (std::string, std::string, double)
- void **LoadEphemerisTrajectory** (std::string, double, std::string)
- void **GoNext** (bool, bool, bool, bool)
- void **CancelTrajectory** (void)
- void **CorrectForStructure** (void)
- void **CorrectForParallactic** (void)
- void **ResetRaDecCumulativeOffset** (void)
- void **UpdateSetup** (bool)
- void **OffsetTrajectoryTipTilt** (double, double)
- void **OffsetTrajectoryRaDec** (double, double, bool)
- void **OffsetTrajectoryRho** (double)
- void **OffsetTrajectoryXY** (double, double, bool)
- void **OffsetTelescope** (double, double, bool, bool)
- void **OffsetFocus** (double)
- void **SetFocus** (double)
- void **SetAnalysisRegion** (std::string, imageRectI)
- void **StartGuiding** (std::string)
- void **StopGuiding** (std::string)
- void **SetFiducial** (std::string, double, double)
- void **ClearFiducial** (std::string)
- void **StartMetrologyLoop** (std::string)
- void **StopMetrologyLoop** (std::string)
- void **ActivateMetrologyLoop** (std::string)
- void **DeactivateMetrologyLoop** (std::string)
- void **CenterProbe** (std::string)
- void **AddCorrections** (std::string, double, double, double, double, double, double)
- void **ResetMetrologyLoop** (std::string)
- void **OffsetProbe** (std::string, double, double)
- void **ForceLoopStateEvent** (std::string)
- void **ForceLegacyStatusEvents** (void)
- void **StopDome** (void)
- void **HomeDome** (void)
- void **CCASDome** (void)
- void **StowDome** (void)
- void **AbortDome** (void)

- void **ResetDome** (void)
- void **MoveDomeTo** (double)
- void **SetDomeVelocityTo** (std::string)
- void **SetDomeDirTo** (std::string)
- void **StopShutter** (void)
- void **OpenShutter** (void)
- void **CloseShutter** (void)
- void **AbortShutter** (void)
- void **ResetShutter** (void)
- void **StopStructure** (void)
- void **CCASStructure** (void)
- void **StowStructure** (void)
- void **MoveStructureToTrueAz** (double)
- void **MoveStructureToStructAz** (double)
- void **ForcePipelineStatusEvent** (std::string)
- void **StartPipeline** (std::string)
- void **StopPipeline** (std::string)
- void **SetTargetTemp** (std::string, double)
- void **SetExposureTime** (std::string, double)
- void **SetIntervalSpeed** (std::string, double)
- void **SetCoAddImages** (std::string, int)
- void **SetBinning** (std::string, int, int)
- void **SetNumFlushes** (std::string, int)
- void **SetCleanImages** (std::string, bool)
- void **SetProcessImages** (std::string, bool)
- void **SetStoreImages** (std::string, bool)
- void **TakeSingleImage** (std::string)
- void **SetProcessingType** (std::string, std::string)
- void **EnableSACDMIMeasurements** (std::string, bool)
- void **EnableCCASDMIMeasurements** (std::string, bool)
- void **SetVirusCCDTemp** (int, double)
- void **ForceUpdateVirusTemps** (void)
- void **SetLRS2CCDTemp** (int, double)
- void **ForceUpdateLRS2Temps** (void)
- void **IssuePFIPControl** (QString, QString, bool)
- void **SendPFIPCommand** (const std::string, Message *)
- void **SetACQCameraFilter** (std::string)
- void **SetACQCameraFocusRel** (double)
- void **SetACQCameraFocusAbs** (double)
- void **SetDitherPosition** (int)
- void **IssueMoveProbes** (QString, QString, QString, QString, QString, QString, QString, QString)
- void **IssueSetProbeGains** (QString, QString, QString, QString, QString, QString, QString, QString)

- void **BeginRangeProbes** (void)
- void **HomeProbes** (void)
- void **EndRangeProbes** (void)
- void **PFIPInterrupt** (void)
- void **DisableProbes** (void)
- void **DeployACQCameraMirror** (void)
- void **RetractACQCameraMirror** (void)

- void **DeployPV_CWFSCameraMirror** (void)
- void **RetractPV_CWFSCameraMirror** (void)
- void **DeployPVCamera** (void)
- void **DeployCWFSCamera** (void)
- void **TurnOnOutletByName** (std::string)
- void **TurnOffOutletByName** (std::string)
- void **TurnOnGroup** (std::string)
- void **TurnOffGroup** (std::string)
- void **Igloo_SetRegister** (std::string, std::string, bool)
- void **PlaySound** (QString, bool)
- void **StopSound** (QString)
- void **ForceHeartbeat** (std::string)
- void **RelayCameraImageCommand** (std::string, std::string)
- void **RelayCameraMetrologyControlCommand** (std::string, std::string)
- void **RelayViewImageExtensionChanged** (std::string, int)
- void **RelaySetViewImageExtension** (std::string, int)

Signals

- void **SubSystemRunning** (std::string, bool)
- void **CommandIssued** (QString, QString)
- void **CommandDone** (QString, bool)
- void **CommandError** (QString, QString)
- void **DashBoardAlert** (std::string, std::string, Logger::LogLevel, fptime, std::string)
- void **IssueCameraImageCommand** (std::string, std::string)
- void **IssueCameraMetrologyControlCommand** (std::string, std::string)
- void **ViewImageExtensionChanged** (std::string, int)
- void **SetViewImageExtension** (std::string, int)
- void **RaDecChanged** (double, double, double, double, double, double, double, double, double)
- void **InstrumentFiducialChanged** (std::string, QPointF)
- void **AnalysisRegionChanged** (std::string, imageRectF)
- void **CorrectionsChanged** (std::string, double, double, double, double, double, double)
- void **MetrologyLoopState** (std::string, bool, bool)
- void **MetrologyDataChanged** (std::string, QPointF, QPointF, QPointF, double, bool, bool, std::string, bool)
- void **NextTrajectoryOccured** (double, double, double, double, double, std::string, std::string, std::string, int)
- void **TrajectoryProgress** (std::string, std::string)
- void **GoNextOccured** (void)
- void **TrajectoryTimes** (double, double)
- void **FocusChanged** (double)
- void **SetupStateChanged** (bool)
- void **TrackerStatusChanged** (void)
- void **TrackerPositionChanged** (void)
- void **TrackerInitProgress** (int, bool, bool, bool, bool)
- void **TrackerInitComplete** (bool)
- void **TrackerMessage** (std::string, std::string)
- void **ProbeStatusChanged** (pfip::ProbeID)
- void **AcqCamStatusChanged** (void)
- void **AcqShutterStatusChanged** (void)

- void **AcqENWAStatusChanged** (void)
- void **AcqMiscStatusChanged** (void)
- void **AcqCWFSStatusChanged** (void)
- void **FCUStatusChanged** (void)
- void **FocalPlaneShutterMoveStart** (int, int)
- void **FocalPlaneShutterMoveEnd** (int, int, bool)
- void **FocalPlaneShutterMove** (int, int)
- void **PFIPMonitorStatusChanged** (void)
- void **VirusMonitorStatusChanged** (int)
- void **DomeStatusChanged** (void)
- void **ShutterStatusChanged** (void)
- void **StructureStatusChanged** (void)
- void **WeatherDataChanged** (void)
- void **DMIStatusChanged** (void)
- void **SAMSStatusChanged** (void)
- void **ImagePipelineStatusChanged** (ImagePipelineStatus)
- void **MetrologyDataChanged** (std::string, double, double, double)
- void **MetrologyDataChanged** (std::string, double, double, double, double, double, double)
- void **MetrologyDataChanged** (std::string, double, double, double, double, double, double, double, double)
- void **APCStartup** (void)
- void **UpdateAPCStatus** (std::string, std::string, int, int, double, int, int)
- void **UpdateAPCGroupStatus** (std::string, std::string)
- void **OCDStatusChanged** (void)
- void **ThermoCubeStatusChanged** (int)
- void **IglooStatusChanged** (void)
- void **ClearVirusTemps** (std::string)
- void **VirusTemps** (std::string, int, int, int, int, int, double, int, double, double, double, int, double, double, double)

Public Member Functions

- **tcsGuiThreadWorker** (const char *name, tcsGui *pParent)
- void **MonitorHeartbeats** (void)
- void **CommandReceiver** (void)
- void **SoundServer** (void)
- void **ForceHeartbeat** (void)
- void **Sound** (void)
- void **RelaySignals** (void)
- void **ReceiveStatusEvents** (void)
- void **ReceiveHeartbeatEvents** (void)
- void **sendTcsAsyncCommand** (const char *cmd, Message *pMsg)
- void **SendTcsAsyncCommands** (void)
- void **ReceiveTcsAsyncReplies** (void)
- void **processTcsAsyncReply** (Message *)
- void **sendTrackerAsyncCommand** (const char *cmd, Message *pMsg)
- void **SendTrackerAsyncCommands** (void)
- void **ReceiveTrackerAsyncReplies** (void)
- void **processTrackerAsyncReply** (Message *)
- void **sendLegacyAsyncCommand** (const char *cmd, Message *pMsg)

- void **SendLegacyAsyncCommands** (void)
- void **ReceiveLegacyAsyncReplies** (void)
- void **processLegacyAsyncReply** (Message ∗)
- void **sendPasAsyncCommand** (const char ∗cmd, Message ∗pMsg)
- void **SendPasAsyncCommands** (void)
- void **ReceivePasAsyncReplies** (void)
- void **processPasAsyncReply** (Message ∗)
- void **sendPfipAsyncCommand** (const char ∗cmd, Message ∗pMsg)
- void **SendPfipAsyncCommands** (void)
- void **ReceivePfipAsyncReplies** (void)
- void **processPfipAsyncReply** (Message ∗)
- void **sendApcAsyncCommand** (const char ∗cmd, Message ∗pMsg)
- void **SendApcAsyncCommands** (void)
- void **ReceiveApcAsyncReplies** (void)
- void **processApcAsyncReply** (Message ∗)
- void **sendVirusAsyncCommand** (const char ∗cmd, Message ∗pMsg)
- void **SendVirusAsyncCommands** (void)
- void **ReceiveVirusAsyncReplies** (void)
- void **processVirusAsyncReply** (Message ∗)
- void **sendLrs2AsyncCommand** (const char ∗cmd, Message ∗pMsg)
- void **SendLrs2AsyncCommands** (void)
- void **ReceiveLrs2AsyncReplies** (void)
- void **processLrs2AsyncReply** (Message ∗)
- void **sendIglooAsyncCommand** (const char ∗cmd, Message ∗pMsg)
- void **SendIglooAsyncCommands** (void)
- void **ReceiveIglooAsyncReplies** (void)
- void **processIglooAsyncReply** (Message ∗)

Protected Member Functions

- void **generateIdAndStartTime** (int &id, string &startTime)
- void **processErrorEvent** (const string &system, const string &source, const string &key, Event ∗pEvent)
- void **processTCSEvent** (const string &source, const string &key, Event ∗pEvent)
- void **processTrackerEvent** (const string &source, const string &key, Event ∗pEvent)
- void **processPFIPEvent** (const string &source, const string &key, Event ∗pEvent)
- void **processPASEvent** (const string &source, const string &key, Event ∗pEvent)
- void **processCamraEvent** (const string &system, const string &source, const string &key, Event ∗pEvent)
- void **processLegacyEvent** (const string &source, const string &key, Event ∗pEvent)
- void **processAPCEvent** (const string &source, const string &key, Event ∗pEvent)
- void **processOCDEvent** (const string &source, const string &key, Event ∗pEvent)
- void **processHPFEvent** (const string &source, const string &key, Event ∗pEvent)
- void **processEPICSRelayEvent** (const string &source, const string &key, Event ∗pEvent)
- void **processIglooEvent** (const string &source, const string &key, Event ∗pEvent)
- void **processTCSStatusEvent** (Event ∗pEvent)

Protected Attributes

- tcsGui * **m_pParent**
- guiSound * **m_pGuiSound**
- bool **m_bTrackerInitError**

The documentation for this class was generated from the following files:

- gui/include/tcsGui.h
- gui/build/moc_tcsGui.cpp
- gui/src/tcsGui.cpp

9.915 TCSHandler Class Reference

Public Member Functions

- **TCSHandler** (TCSDriver *d, const MessageHandlerDescriptor &m)

Public Attributes

- TCSDriver * **driver**

The documentation for this class was generated from the following file:

- tcs/tcs_handlers.h

9.916 TCSLog::TCSLog Class Reference

Public Member Functions

- def __init__
- def **strip_end**
- def **log**
- def **log_debug**
- def **log_info**
- def **log_error**
- def **log_warn**
- def **log_fatal**
- def **log_alarm**
- def **__exit__**

Static Public Attributes

- int **Off** = 0
- int **Debug** = 1
- int **Info** = 2
- int **Warn** = 4
- int **Error** = 8
- int **Fatal** = 16
- int **Alarm** = 32

9.916.1 Member Function Documentation

9.916.1.1 def TCSLog::TCSLog::__init__ (*self*, *sourceName* = `None`, *logRelayRoute* = `None`, *namedRoute* = `None`)

```
Create an Iinstance of a TCSLog object.

The sourceName cannot be empty and should identify the the script or program
doing the logging.

The logRelayRoute is the route to the log relay server.
The namedRoute is the route to the naemed server.

If both are left empty, logging will be restricted to the console.
If no relay route is given, but named is, the relay route will be
looked up using named. If a relay route is given, it will be used as is.
```

The documentation for this class was generated from the following file:

- scripting/pytcs/build/lib.linux-x86_64-2.7/TCSLog.py

9.917 pytcs::TCSLog::TCSLog Class Reference

Public Member Functions

- def __init__
- def **strip_end**
- def **log**
- def **log_debug**
- def **log_info**
- def **log_error**
- def **log_warn**
- def **log_fatal**
- def **log_alarm**
- def **__exit__**

Static Public Attributes

- int **Off** = 0
- int **Debug** = 1
- int **Info** = 2
- int **Warn** = 4
- int **Error** = 8
- int **Fatal** = 16
- int **Alarm** = 32

9.917.1 Member Function Documentation

9.917.1.1 def pytcs::TCSLog::TCSLog::__init__ (*self*, *sourceName* = `None`, *logRelayRoute* = `None`, *namedRoute* = `None`)

```
Create an Iinstance of a TCSLog object.

The sourceName cannot be empty and should identify the the script or program
doing the logging.

The logRelayRoute is the route to the log relay server.
The namedRoute is the route to the naemed server.

If both are left empty, logging will be restricted to the console.
If no relay route is given, but named is, the relay route will be
looked up using named. If a relay route is given, it will be used as is.
```

The documentation for this class was generated from the following file:

- scripting/pytcs/TCSLog.py

9.918 pytcs::TCSNamed::TCSNamed Class Reference

Public Member Functions

- def **__init__**
- def **lookup**

The documentation for this class was generated from the following file:

- scripting/pytcs/TCSNamed.py

9.919 TCSNamed::TCSNamed Class Reference

Public Member Functions

- def **__init__**
- def **lookup**

The documentation for this class was generated from the following file:

- scripting/pytcs/build/lib.linux-x86_64-2.7/TCSNamed.py

9.920 TCSServerConfig Class Reference

Public Member Functions

- **TCSServerConfig** (bool isNamed=0)
- virtual void print_summary ()

  Called prior to displaying options.

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- double **zero_offset** [6]
- bool **disable_mount_model**
- int **dmi_head**

9.920.1 Member Function Documentation

9.920.1.1 virtual void TCSServerConfig::print_summary () [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.920.1.2 virtual void TCSServerConfig::set_options (OptionParser & *optionParser*) [inline, virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

The documentation for this class was generated from the following file:

- tcs/tcs_server_config.h

9.921 TCSSocket Class Reference

Public Member Functions

- void **setDebug** (bool state=true)
- bool connect (std::string address, int retries=2, int timeSec=5)

  Create socket client.

- bool transmit (unsigned char *buf, unsigned int length)

  Transmit data.

- int receive (unsigned char *buf, unsigned int length, bool(*checkDone)(unsigned char *, unsigned int, unsigned int)=NULL, int timeout_ms=-1)

  Receive data.

- void flush (void)

  Empty socket.

- bool connected (void)

  Return connection state.

- void disconnect (void)

  Close socket.

Protected Member Functions

- bool **connect** (struct addrinfo *result, int timeoutSec)
- bool **writeSocket** (unsigned char *buf, unsigned int length)
- int **readSocket** (unsigned char *buf, unsigned int length, bool(*checkDone)(unsigned char *, unsigned int, unsigned int), int timeout_ms)

Protected Attributes

- bool **m_debug**
- bool m_connected

  Connection state.

- int m_defaultReadTimeout

  Read timeout if not specified.

- int m_socketHandle

  socket handle

- std::string **m_address**
- std::string **m_host**
- std::string **m_port**
- char **m_hostName** [80]
- char **m_servName** [80]

9.921.1 Member Function Documentation

9.921.1.1 bool TCSSocket::connect (std::string *address*, int *retries* = 2, int *timeSec* = 5)

Create a socket client and connect to a remote host and report error condition.

Parameters:

address [in] - IP address or host name and port of server ("localhost:port")

9.921.1.2 bool TCSSocket::connected (void) [inline]

Return connection state.

9.921.1.3 void TCSSocket::disconnect (void)

Close socket and mark socket disconnected

9.921.1.4 void TCSSocket::flush (void)

Empty socket of any un-received data.

9.921.1.5 int TCSSocket::receive (unsigned char * *buf*, unsigned int *length*, bool(*)(unsigned char *, unsigned int, unsigned int) *checkDone* = NULL, int *timeout_ms* = -1)

Receive data using an optional call back function to check for the end of the transmission.

Parameters:

buffer [in] - Address of buffer to receive data

length [in] - Length of buffer

checkDone [in] - Optional address of function that receive will call to check for end of transmission

timeout_ms [in] - Optional time out in milliseconds

Returns:

>0 - Length of data received
0 - socket closed by peer
-1 -socket error
-2 - timed out

9.921.1.6 bool TCSSocket::transmit (unsigned char * *buf*, unsigned int *length*)

Transmit data and report error condition.

Parameters:

buffer [in] - Address of buffer to send

length [in] - Length of buffer

The documentation for this class was generated from the following files:

- het_lib/common/include/tcs_socket.h
- het_lib/common/src/tcs_socket.cpp

9.922 pytcs::tcssubsystem::TCSSubSystem Class Reference

Public Member Functions

- def **__init__**
- def **help**
- def **wait**
- def **__exit__**

Public Attributes

- **name**
- **api**
- **outstanding_**

The documentation for this class was generated from the following file:

- scripting/pytcs/tcssubsystem.py

9.923 TCSSubSystem Class Reference

9.923.1 Detailed Description

A base class for handling TCS applications/processes. Together with the TCS_App class, it performs common functions of the TCS subsystems.

`#include <tcs_subsystem.h>`

Public Member Functions

- **TCSSubSystem** (const string &appName, TCSConfig *pConfig)
- virtual bool **init** (void)
- virtual int **start** (void)=0
- virtual void **stop** (void)

Public Attributes

- string **appName_**

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_subsystem.h

9.924 tcssubsystem::TCSSubSystem Class Reference

Public Member Functions

- def **__init__**
- def **help**
- def **wait**
- def **__exit__**

Public Attributes

- **name**
- **api**
- **outstanding_**

The documentation for this class was generated from the following file:

- scripting/pytcs/build/lib.linux-x86_64-2.7/tcssubsystem.py

9.925 liborchestrate::TcsSubsystem Class Reference

Public Member Functions

- def **__init__**

The documentation for this class was generated from the following file:

- integration/liborchestrate.py

9.926 TCSTelnet Class Reference

Public Member Functions

- virtual bool **sendCmdCheckReply** (int cmd, char ∗cmdString, bool(∗checkDone)(unsigned char ∗, unsigned int, unsigned int), const char ∗terminators="\n", unsigned int timeout_sec=5)
- virtual bool **checkReply** (int, unsigned int, unsigned char ∗)
- void **setEmptyBeforeNextCmd** (void)
- void **setLastCmdError** (const std::string &error)
- std::string **getLastCmdError** ()

The documentation for this class was generated from the following files:

- het_lib/common/include/tcs_telnet.h
- het_lib/common/src/tcs_telnet.cpp

9.927 telescope Struct Reference

Structure containing information about the HET.

`#include <point.h>`

Public Attributes

- int scope_
  Arbitrary number.
- char * nameShort_
  Abbreviation (in capitals).
- char * name_
  Full name.
- double longitude_
  (radians, West +ve)
- double geolat_
  geodetic latitude (radians, North +ve)
- double height_
  height above sea level (metres)

The documentation for this struct was generated from the following file:

- integration/sir/Het_trajectory/point.h

9.928 TCSClient::TelescopeData Struct Reference

Public Attributes

- bool **instrumentDataValid**
- double **current_x**
- double **current_y**
- double **current_orientation_deg**
- double **current_arm_angle_deg**
- double **current_probe_angle_deg**
- double **target_x**
- double **target_y**
- double **target_ra**
- double **target_dec**
- double **target_eq**
- double **inst_ra_2000_hrs**
- double **inst_dec_2000_deg**
- double **inst_eq_2000**
- bool **pointingDataValid**
- double **ra_hrs**
- double **dec_deg**
- double **ra_2000_hrs**
- double **dec_2000_deg**
- double **az_deg**
- double **el_deg**
- double **parallactic_angle_deg**
- double **hour_angle**
- double **zenith_distance_deg**
- double **lst_hrs**
- itf_coordinate **itf**
- double **rho_offset_deg**
- bool **dmiDataValid**
- bool **dmi_validityError**
- double **dmi_distance**
- double **imageDateTime**

The documentation for this struct was generated from the following file:

- pas/include/tcs_client.h

9.929 Test Class Reference

Start/Stop the dome movement test.

9.929.1 Detailed Description

Parameters:

state <string> - "Start" or "Stop"

`#include <legacyDomeSubSystem.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacyDomeSubSystem.h

9.930 test Class Reference

Public Member Functions

- **test** (Archon &d, const MessageHandlerDescriptor &m)
- void process (Message *message)

Public Attributes

- int **nsteps** = 2500000
- tuple **draws** = np.random.randint(0,2,size=nsteps)
- tuple **steps** = np.where(draws>0,1,-1)
- tuple **walk** = steps.cumsum()

9.930.1 Member Function Documentation

9.930.1.1 void test::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following files:

- ics/archon.h
- scripting/scripts/VE2_Temperatures/test.py

9.931 ThermoCubeStatus Class Reference

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- struct ThermoCubeStatus::status **m_status**

The documentation for this class was generated from the following files:

- gui/include/thermoCubeStatusData.h
- gui/src/thermoCubeStatusData.cpp

9.932 tthread::thread Class Reference

Thread class.

`#include <tinythread.h>`

Classes

- class id

Public Member Functions

- thread ()
- template<typename F >
  thread (F f)
- template<typename F , typename A >
  thread (F f, A a)
- template<typename 173. 14463331>
  thread (173.14463331 *object, void(173.14463331::*function)())
- template<typename 173. 14463331, typename A >
  thread (173.14463331 *object, void(173.14463331::*function)(A), A arg)
- virtual ~thread ()
- virtual void execute ()

  Create and start executing the thread.

- void join ()
- bool joinable () const
- void detach ()
- id get_id () const

  Return the thread ID of a thread object.

- void kill (int sig)

  Send a signal to a thread.

- pthread_t native_handle ()
- **thread** (const thread &)
- thread & **operator=** (const thread &)

Static Public Member Functions

- static unsigned hardware_concurrency ()

Protected Member Functions

- virtual void set_trampoline ()
- void finish_init_thread ()

  Common part of thread initialization.

Static Protected Member Functions

- static void * thread_execute_once (void *arg)

Protected Attributes

- pthread_t mHandle

  Thread handle.

- mutex mDataMutex

  Serializer for access to the thread private data.

- bool mNotAThread

  True if this object is not a thread of execution.

- _thread_start_info * ti_

  Trampoline object to allow threading methods, functions, and functors.

9.932.1 Constructor & Destructor Documentation

9.932.1.1 tthread::thread::thread () [inline]

Default constructor. Construct a `thread` object without an associated thread of execution (i.e. non-joinable).

9.932.1.2 template<typename F > tthread::thread::thread (F *f*) [inline]

Thread starting constructor. Construct a `thread` object with a new thread of execution.

Parameters:

←*f* A function pointer to a function of type: `void fun(void)` or a Functor object with `operator() (void)`

Note:

This constructor is not fully compatible with the standard C++ thread class. It is more similar to the pthread_create() (POSIX) and CreateThread() (Windows) functions.

9.932.1.3 template<typename F , typename A > tthread::thread::thread (F *f*, A *a*) [inline]

Thread starting constructor. Construct a `thread` object with a new thread of execution.

Parameters:

←*f* A function pointer to a function of type: `void fun(void * arg)` or a Functor object with `operator() (Arg& arg)`

←*aArg* Argument to the thread function or functor.

Note:

This constructor is not fully compatible with the standard C++ thread class. It is more similar to the pthread_create() (POSIX) and CreateThread() (Windows) functions.

9.932.1.4 template<typename 173. 14463331> tthread::thread::thread (173.14463331 * *object*, void(173.14463331::*)() *function*) [inline]

Thread starting constructor. Construct a `thread` object with a new thread of execution.

Parameters:

← ***object*** pointer to an instance of class `C`

← ***function*** pointer to a memeber function of `C` `void C::function(void)`

Note:

This constructor is not fully compatible with the standard C++ thread class. It is more similar to the pthread_create() (POSIX) and CreateThread() (Windows) functions.

9.932.1.5 template<typename 173. 14463331, typename A > tthread::thread::thread (173.14463331 * *object*, void(173.14463331::*)(A) *function*, A *arg*) [inline]

Thread starting constructor. Construct a `thread` object with a new thread of execution.

Parameters:

← ***object*** pointer to an instance of class `C`

← ***function*** pointer to a memeber function of `C` with one argument of type `A` `void C::function(A arg)`

Note:

This constructor is not fully compatible with the standard C++ thread class. It is more similar to the pthread_create() (POSIX) and CreateThread() (Windows) functions.

9.932.1.6 virtual tthread::thread::~thread () [inline, virtual]

Destructor.

Note:

If the thread is joinable upon destruction, `std::terminate()` will be called, which terminates the process. It is always wise to do `join()` before deleting a thread object.

9.932.2 Member Function Documentation

9.932.2.1 void tthread::thread::detach () [inline]

Detach from the thread. After calling `detach()`, the thread object is no longer assicated with a thread of execution (i.e. it is not joinable). The thread continues execution without the calling thread blocking, and when the thread ends execution, any owned resources are released.

9.932.2.2 unsigned tthread::thread::hardware_concurrency () [static]

Determine the number of threads which can possibly execute concurrently. This function is useful for determining the optimal number of threads to use for a task.

Returns:

The number of hardware thread contexts in the system.

Note:

If this value is not defined, the function returns zero (0).

9.932.2.3 void tthread::thread::join () [inline]

Wait for the thread to finish (join execution flows). After calling `join()`, the thread object is no longer associated with a thread of execution (i.e. it is not joinable, and you may not join with it nor detach from it).

9.932.2.4 bool tthread::thread::joinable () const [inline]

Check if the thread is joinable. A thread object is joinable if it has an associated thread of execution.

9.932.2.5 pthread_t tthread::thread::native_handle () [inline]

Get the native handle for this thread.

Note:

Under Windows, this is a `HANDLE`, and under POSIX systems, this is a `pthread_t`.

9.932.2.6 virtual void tthread::thread::set_trampoline () [inline, protected, virtual]

Set the pthread-compatible trampoline function that is executed in a new thred. This function may be overloaded in derived classes to implement different scheduling policies, for example periodic scheduling instead of just once.

Reimplemented in tthread::ThreadWorker.

9.932.2.7 void * tthread::thread::thread_execute_once (void * *arg*) [static, protected]

This is the pthread-compatible function that is executed in a separate thread. The `void*` argument is a this-pointer to the ThreadWorker instance.

The documentation for this class was generated from the following files:

- het_lib/tthread/include/tinythread.h
- het_lib/tthread/src/tinythread.cpp

9.933 tthread::ThreadPool Class Reference

9.933.1 Detailed Description

Thread executing quanta of recurring work with some priod, with the ability to arbitrarily pause and restart periodic execution.

`#include <threadpool.h>`

Public Types

- enum { **stopped_** = 0, **started_** = 1 }

Public Member Functions

- **ThreadPool** (const int max_pool_size=16)
- int **destroy_threadpool** ()
- void * **execute_thread** ()
- pthread_t **add_task** (_thread_start_info *task)

Protected Member Functions

- int **add_thread_to_pool** (void)

Protected Attributes

- volatile int **pool_state_**
- int **max_pool_size_**
- mutex **task_mutex_**
- std::vector< ThreadPoolWorker > **threads_**
- std::deque< _thread_start_info * > **tasks_**

The documentation for this class was generated from the following files:

- het_lib/tthread/include/threadpool.h
- het_lib/tthread/src/threadpool.cpp

9.934 tthread::ThreadPoolException Class Reference

Public Member Functions

- **ThreadPoolException** (const char *format,...)
- virtual const char * **what** () const throw ()

The documentation for this class was generated from the following file:

- het_lib/tthread/include/threadpool.h

9.935 tthread::ThreadPoolWorker Struct Reference

Public Attributes

- volatile bool **busy_**
- pthread_t **pthread_**
- condition_variable **cond_**

The documentation for this struct was generated from the following file:

- het_lib/tthread/include/threadpool.h

9.936 threadsafe_deque< Entry > Class Template Reference

9.936.1 Detailed Description

template<typename Entry> class threadsafe_deque< Entry >

Thread-safe queue

`#include <tsdeque.h>`

Public Member Functions

- void **push_back** (const Entry &data)
- void **push_front** (const Entry &data)
- bool **empty** ()
- size_t **size** ()
- Entry **wait_and_pop** ()
- bool **wait_for_and_pop** (Entry &e, double timeout, double *spent)
- void **erase** (const Entry &entry)
- void **clear** (void)

The documentation for this class was generated from the following file:

- het_lib/tthread/include/tsdeque.h

9.937 threadsafe_list< Entry > Class Template Reference

9.937.1 Detailed Description

template<typename Entry> class threadsafe_list< Entry >

Thread-safe (paritially) list class.

`#include <tslist.h>`

Public Member Functions

- void **push_front** (const Entry &data)
- void **push_back** (const Entry &data)
- bool **empty** ()
- size_t **size** ()
- Entry **wait_and_pop_front** ()
- void **remove** (const Entry &entry)
- void **erase** (const Entry &entry)
- void **clear** (void)

The documentation for this class was generated from the following file:

- het_lib/tthread/include/tslist.h

9.938 threadsafe_queue< Entry > Class Template Reference

9.938.1 Detailed Description

template<typename Entry> class threadsafe_queue< Entry >

Thread-safe queue

`#include <tsqueue.h>`

Public Member Functions

- void **push** (const Entry &data)
- bool **empty** () const
- size_t **size** () const
- bool **try_pop** (Entry &entry)
- Entry & **wait_and_pop** ()

The documentation for this class was generated from the following file:

- het_lib/tthread/include/tsqueue.h

9.939 tthread::ThreadWorker Class Reference

9.939.1 Detailed Description

Thread executing quanta of recurring work with some priod, with the ability to arbitrarily pause and restart periodic execution.

The object being threaded can be anything that `tthread` accepts.

The execution is periodic, starting every given number of seconds. If the processing takes longer than one period, the function is executed as often as possible, i.e. is rescheduled immediately after finishing. Otherwise an appropriate amount of time is spent sleeping.

Execution begins upon calling `run()` and stops upon calling `stop()`, but the current iteration is allowed to finish and return. Execution can be resumed later by calling `run()` again.

`#include <threadworker.h>`

Public Member Functions

- template<typename F >
  ThreadWorker (F f, const double period=0.0)
- template<typename F , typename A >
  ThreadWorker (F f, A a, const double period=0.0)
- template<typename 173. 14463331>
  ThreadWorker (173.14463331 ∗object, void(173.14463331::∗function)(), const double period=0.0)
- template<typename 173. 14463331, typename A >
  ThreadWorker (173.14463331 ∗object, void(173.14463331::∗function)(A), A arg, const double period=0.0)
- void run ()
- void stop ()
- void **exit** ()
- bool is_running (void) const
- void set_period (const double new_period)

Protected Member Functions

- virtual void set_trampoline ()

Static Protected Member Functions

- static void ∗ thread_execute_periodic (void ∗arg)

Protected Attributes

- double period_

  period of repeated execution in seconds

- bool run_

  state of thread.

- bool first_run_

  first time we're run?

- volatile bool exit_

  signal to exit the thread when true. cannot be restarted.

- mutex m_

  mutex for synchronization.

- condition_variable c_

  condition variable to pause execution while !running.

9.939.2 Constructor & Destructor Documentation

9.939.2.1 template<typename F > tthread::ThreadWorker::ThreadWorker (F *f*, const double *period* = 0.0) [inline]

Periodic thread starting constructor. Construct a `WorkerThread` object with a new thread of execution.

Parameters:

← ***f*** A function pointer to a function of type: `void fun(void)` or a Functor object with `operator() (void)`

← ***period*** The period of exectution in seconds

9.939.2.2 template<typename F , typename A > tthread::ThreadWorker::ThreadWorker (F *f*, A *a*, const double *period* = 0.0) [inline]

Periodic thread starting constructor. Construct a `thread` object with a new thread of execution.

Parameters:

← ***f*** A function pointer to a function of type: `void fun(void * arg)` or a Functor object with `operator() (Arg& arg)`

← ***a*** Argument to the thread function or functor.

← ***period*** The period of exectution in seconds

9.939.2.3 template<typename 173. 14463331> tthread::ThreadWorker::ThreadWorker (173.14463331 * *object*, void(173.14463331::*)() *function*, const double *period* = 0.0) [inline]

Periodic thread starting constructor. Construct a `thread` object with a new thread of execution.

Parameters:

← ***object*** pointer to an instance of class `C`

← ***function*** pointer to a memeber function of `C` `void C::function(void)`

← ***period*** The period of exectution in seconds

9.939.2.4 template<typename 173. 14463331, typename A > tthread::ThreadWorker::ThreadWorker (173.14463331 * *object*, void(173.14463331::*)(A) *function*, A *arg*, const double *period* = 0.0) [inline]

Periodic thread starting constructor. Construct a `thread` object with a new thread of execution.

Parameters:

← ***object*** pointer to an instance of class `C`

← ***function*** pointer to a memeber function of `C` with one argument of type `A` `void C::function(A arg)`

← ***period*** The period of exectution in seconds

9.939.3 Member Function Documentation

9.939.3.1 bool tthread::ThreadWorker::is_running (void) const [inline]

Query the state of the thread. Note that this can return `false` despite the current iteration of `do_quantum_of_work` still running, indicating a stop has been requested but not yet reached.

9.939.3.2 void tthread::ThreadWorker::run () [inline]

Start repeated execution of the `do_quantum_of_work` function. Execution continues until `stop()` is called.

9.939.3.3 void tthread::ThreadWorker::set_period (const double *new_period*) [inline]

set a new period in seconds.

9.939.3.4 virtual void tthread::ThreadWorker::set_trampoline () [inline, protected, virtual]

overload the trampoline function with one that implements our periodic execution

Reimplemented from tthread::thread.

9.939.3.5 void tthread::ThreadWorker::stop (void) [inline]

Request stopping repeated execution of the `do_quantum_of_work` function after the current iteration finishes. Execution resumes when `run()` is called.

9.939.3.6 void * tthread::ThreadWorker::thread_execute_periodic (void * *arg*) [static, protected]

This is the pthread-compatible function that is executed in a separate thread. It checks for the condition and repeatedly calls the user function of the subclass.

The `void*` argument is a this-pointer to the ThreadWorker instance.

The documentation for this class was generated from the following files:

- het_lib/tthread/include/threadworker.h
- het_lib/tthread/src/threadworker.cpp

9.940 TimeoutHandler Class Reference

Public Member Functions

- virtual void process (Message ∗message)

9.940.1 Member Function Documentation

9.940.1.1 virtual void TimeoutHandler::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- het_lib/messaging/include/builtin_handlers.h

9.941 TimeoutNextMove Class Reference

9.941.1 Detailed Description

Cause the next probe move to timeout.

Optional parameters:

timeout<string> - Length of time in seconds before timeout after move call (default is 10).

`#include <pfip_handler_doc.h>`

Public Member Functions

- **TimeoutNextMove** (Receiver *rcv, ACQ_SubSystem *acq)
- virtual void HandlerAction (Message *pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message *pMessage)

  Cleanup after performing handler action.

The documentation for this class was generated from the following files:

- pfip/include/pfipServer.h
- pfip/src/ProbeAssemblyHandlers.cpp

9.942 tthread::Timer Class Reference

9.942.1 Detailed Description

simple timer class

`#include <threadworker.h>`

Public Member Functions

- void **start** ()
- void **stop** ()
- double **elapsedSeconds** ()

The documentation for this class was generated from the following file:

- het_lib/tthread/include/threadworker.h

9.943 tracker_handlers::timeToIntercept Class Reference

Public Member Functions

- **timeToIntercept** (TrackerDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.943.1 Member Function Documentation

9.943.1.1 void tracker_handlers::timeToIntercept::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tracker/tracker_handlers.h

9.944 timeToIntercept Class Reference

9.944.1 Detailed Description

Reports the time to intercept an initial trajectory vector. Sends a tmcs_move_time command pair to TMCS and responds with the time the tracker would take to move from the current position to the first point of the point pair given assuming they represent an initial trajectory vector.

Required parameters:

id1<int> - Trajectory ID of the first point

t1<double> - Index time of the first point

x1<double> - x coordinate of the first point (mm)

y1<double> - y coordinate of the first point (mm)

z1<double> - z coordinate of the first point (mm)

rho1<double> - rho coordinate of the first point (deg)

theta1<double> - theta coordinate of the first point (deg)

phi1<double> - phi coordinate of the first point (deg)

id2<int> - Trajectory ID of the second point

t2<double> - Index time of the second point

x2<double> - x coordinate of the second point (mm)

y2<double> - y coordinate of the second point (mm)

z2<double> - z coordinate of the second point (mm)

rho2<double> - rho coordinate of the second point (deg)

theta2<double> - theta coordinate of the second point (deg)

phi2<double> - phi coordinate of the second point (deg)

`#include <tracker_handler_doc.h>`

The documentation for this class was generated from the following file:

- tracker/tracker_handler_doc.h

9.945 tle_t Struct Reference

Public Attributes

- double **epoch**
- double **xndt2o**
- double **xndd6o**
- double **bstar**
- double **xincl**
- double **xnodeo**
- double **eo**
- double **omegao**
- double **xmo**
- double **xno**
- int **norad_number**
- int **bulletin_number**
- int **revolution_number**
- char **classification**
- char **ephemeris_type**
- char **intl_desig** [9]

The documentation for this struct was generated from the following file:

- tcs/traj_comp/NORADlib/norad.h

9.946 TMCSState Class Reference

9.946.1 Detailed Description

XXX

`#include <tmcs_api.h>`

Public Member Functions

- **TMCSState** (TrackerConfig *c)
- void **set_status** (const std::string &status)
- void **process_ack** (const std::string &status)
- void **process_response** (const std::string &status)
- void **process_connected** (bool connected)
- void **process_abort** (void)
- void **process_msg** (const std::string &status)
- void **clear** (int idx)

The documentation for this class was generated from the following files:

- tracker/tmcs_api.h
- tracker/tmcs_api.cpp

9.947 liborchestrate::TmcsSubsystem Class Reference

Public Member Functions

- def **__init__**

The documentation for this class was generated from the following file:

- integration/liborchestrate.py

9.948 toggle Class Reference

Public Member Functions

- **toggle** (IglooSensors &d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.948.1 Member Function Documentation

9.948.1.1 void toggle::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- igloo_sensors/igloo_sensors.h

9.949 toggle_gui_token Class Reference

9.949.1 Detailed Description

Toggles the status of the master GUI token. Once acquired, the same token must be provided to a subsequent call to release the token.

Required parameters:

token<string> - The token requested or released.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **toggle_gui_token** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.949.2 Member Function Documentation

9.949.2.1 void toggle_gui_token::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/tcs_handlers.h

9.950 TopConfig Class Reference

Public Member Functions

- **TopConfig** (bool isNamed=0)

Public Attributes

- string **systems** [64]
- string **key**

The documentation for this class was generated from the following file:

- top/tcs_top_config.h

9.951 TopDriver Class Reference

Public Member Functions

- **TopDriver** (const string &appName, TopConfig *config)
- int **start** ()
- void **stop** ()
- void **process** (Message *m)
- void **event_thread** (void)

The documentation for this class was generated from the following files:

- top/tcs_top_driver.h
- top/tcs_top_driver.cpp

9.952 tracCoord Struct Reference

Structure for internal coordinates.

`#include <point.h>`

Public Attributes

- double **ra**
- double **dec**
- double **ha**
- double **mra**
- double **mdec**
- double **ara**
- double **adec**

The documentation for this struct was generated from the following files:

- integration/sir/Het_trajectory/point.h
- tcs/traj_comp/point.h

9.953 tracker_abort Class Reference

9.953.1 Detailed Description

Sends an ABORT command to the tracker hardware, aborting an active trajectory or move.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **tracker_abort** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.953.2 Member Function Documentation

9.953.2.1 void tracker_abort::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/tcs_handlers.h

9.954 tracker_coordinate Class Reference

9.954.1 Detailed Description

Class representing a point in the tracker frame, either ideal or real.

`#include <tcs_coordinates.h>`

Public Member Functions

- **tracker_coordinate** (const fptime &t, const double &x, const double &y, const double &z, const double &rho, const double &theta, const double &phi)
- **tracker_coordinate** (const tracker_coordinate &t)
- tracker_coordinate & **operator+=** (const tracker_coordinate &p)
- tracker_coordinate & **operator-=** (const tracker_coordinate &p)
- tracker_coordinate & **operator=** (const tracker_coordinate &p)
- tracker_coordinate **operator+** (const tracker_coordinate &p)
- tracker_coordinate **operator-** (const tracker_coordinate &p)
- tracker_coordinate **operator**∗ (const double &d)

Public Attributes

- fptime **t_**
- double **x_**
- double **y_**
- double **z_**
- double **rho_**
- double **theta_**
- double **phi_**

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_coordinates.h

9.955 tracker_init Class Reference

9.955.1 Detailed Description

Sends an INIT command to the tracker hardware, initiating the initialization process for the tracker.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **tracker_init** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.955.2 Member Function Documentation

9.955.2.1 void tracker_init::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/tcs_handlers.h

9.956 tracker_move_angular Class Reference

9.956.1 Detailed Description

Moves the tracker to the given angular coordinates on sky.

Required parameters:

x<double> - Target telecentric x position. (arcsecond)

y<double> - Target telecentric y position. (arcsecond)

type<string> - The type of move requested, rel or abs, relative or absolute respectively.

vel<string> - The velocity with which to move: SLEW_SLOW, SLEW_FAST, TRK_SLOW or TRK_-FAST.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **tracker_move_angular** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.956.2 Member Function Documentation

9.956.2.1 void tracker_move_angular::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/tcs_handlers.h

9.957 tracker_move_itf Class Reference

9.957.1 Detailed Description

Moves the tracker to the given ITF coordinate.

Required parameters:

x<double> - Target x position, in ITF. (mm)

y<double> - Target y position, in ITF. (mm)

z<double> - Target z position, in ITF. (mm)

rho<double> - Target rho position, in ITF. (degree)

theta<double> - Target theta position, in ITF. (degree)

phi<double> - Target phi position, in ITF. (degree)

type<string> - The type of move requested, rel or abs, relative or absolute respectively.

vel<string> - The velocity with which to move: SLEW_SLOW, SLEW_FAST, TRK_SLOW or TRK_-FAST.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **tracker_move_itf** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.957.2 Member Function Documentation

9.957.2.1 void tracker_move_itf::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/tcs_handlers.h

9.958 tracker_move_tsf Class Reference

9.958.1 Detailed Description

Moves the tracker to the given TSF coordinate.

Required parameters:

x<double> - Target x position, in TSF. (mm)

y<double> - Target y position, in TSF. (mm)

z<double> - Target z position, in TSF. (mm)

rho<double> - Target rho position, in TSF. (degree)

theta<double> - Target theta position, in TSF. (degree)

phi<double> - Target phi position, in TSF. (degree)

type<string> - The type of move requested, rel or abs, relative or absolute respectively.

vel<string> - The velocity with which to move: SLEW_SLOW, SLEW_FAST, TRK_SLOW or TRK_-FAST.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **tracker_move_tsf** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.958.2 Member Function Documentation

9.958.2.1 void tracker_move_tsf::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/tcs_handlers.h

9.959 tracker_position Class Reference

9.959.1 Detailed Description

Returns the current tracker position.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **tracker_position** (TCSDriver ∗d, const MessageHandlerDescriptor &m)
- void process (Message ∗message)

9.959.2 Member Function Documentation

9.959.2.1 void tracker_position::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/tcs_handlers.h

9.960 GLTrackerDriver::tracker_position Struct Reference

Public Member Functions

- double **distance_to** (const tracker_position &other)

Public Attributes

- double **x**
- double **y**
- double **z**
- double **theta**
- double **phi**
- double **rho**

The documentation for this struct was generated from the following file:

- playground/gltracker/gltracker_driver.h

9.961 TrackerClient Class Reference

Public Member Functions

- **TrackerClient** (const std::string &route, MountModel &model, TCSDriver *driver)
- void **set_buffer_size** (int size)
- bool **load_point** (int trajectory_id, const ShotPoint &point, bool broadcast_point=0)
- bool **adjust_trajectory_for_current_position** (Trajectory &t)
- bool **clear_error** (int cmd_index)
- void **update_status** (Event *e)
- bool **on_trajectory** (void)
- double **get_move_timeout** (const std::string &velocity)
- bool **is_abort_event** (Event *e)
- bool **init** (bool do_exception=1)
- bool **abort** (bool do_exception=1)
- bool **move_itf** (const std::string &type, const std::string &vel, double x, double y, double z, double rho, double theta, double phi, bool do_exception=1)
- bool **move_tsf** (const std::string &type, const std::string &vel, double x, double y, double rho, double hex1, double hex2, double hex3, double hex4, double hex5, double hex6, bool do_exception=1)
- bool **move_angular** (const std::string &type, const std::string &vel, double x, double y, bool do_exception=1)
- TrackerPosition **get_position** (void)
- bool **check_itf_range** (fptime t, double x, double y, double z, double rho, double theta, double phi)
- void **wait_for_offset** (void)

The documentation for this class was generated from the following files:

- tcs/tracker_client.h
- tcs/tracker_client.cpp

9.962 TrackerConfig Class Reference

Public Member Functions

- **TrackerConfig** (bool isNamed=0)
- void print_summary (void)

  Called prior to displaying options.

- void set_options (OptionParser &optionParser)

  Adds options to default set.

Public Attributes

- string **tmcs_ip_address**
- int **tmcs_server_port**
- int **tmcs_log_port**
- int **tracker_server_port**
- bool **relay_log_messages**
- bool **disable_timeouts**

9.962.1 Member Function Documentation

9.962.1.1 void TrackerConfig::print_summary (void) [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.962.1.2 void TrackerConfig::set_options (OptionParser & *optionParser*) [inline, virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

The documentation for this class was generated from the following file:

- tracker/tracker_config.h

9.963 TrackerData Class Reference

Public Attributes

- float **x**
- float **y**
- float **d**
- float **xr**
- float **yr**
- float **scale**
- float **f**

The documentation for this class was generated from the following file:

- het_lib/wfs_lib/include/wfslib.h

9.964 TrackerDriver Class Reference

9.964.1 Detailed Description

Tracker "driver" class that encapsulate all interfaces to the tracker hardware.

`#include <tracker_driver.h>`

Public Member Functions

- **TrackerDriver** (const string &appName, TrackerConfig *config)
- int **start** ()
- void **stop** ()
- void **initTMCS** (void)
- void **setTraj** (int tSetNum, tracker_coordinate rtf)
- void **abortMove** (int tset_num=0)
- void **clearError** (int msg_idx)
- void **moveRTF** (string type, string vel, const rtf_coordinate &rtf)
- void **moveTSF** (string type, string vel, const tsf_coordinate &tsf)
- fptime **timeToIntercept** (int id1, const tracker_coordinate &rtf1, int id2, const tracker_coordinate &rtf2)
- void **setBufferSize** (int size)
- void tracker_server (void)
- void tracker_client (void)
- void **tracker_logs** (void)

9.964.2 Member Function Documentation

9.964.2.1 void TrackerDriver::tracker_client (void)

A thread method that will serve as the watchdog ACK sender, the client of TMCS.

9.964.2.2 void TrackerDriver::tracker_server (void)

Server method that runs in a separate thread and maps received messages onto TrackerDriver instance state.

The documentation for this class was generated from the following files:

- tracker/tracker_driver.h
- tracker/tracker_driver.cpp

9.965 TrackerException Class Reference

Public Member Functions

- **TrackerException** (tcsexcept::tcs_exception_t __tracker, const std::string &w, const char ∗file, int line, const char ∗func)

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_exception.h

9.966 TrackerHandler Class Reference

Base class for all remote interfaces.

9.966.1 Detailed Description

The TrackerHandler base class provides common functionality used by all Tracker Handlers.

`#include <tracker_handlers.h>`

Public Member Functions

- **TrackerHandler** (TrackerDriver *d, const MessageHandlerDescriptor &m)

Public Attributes

- TrackerDriver * driver

  Pointer binding the handler to an instance of the TrackerDriver.

The documentation for this class was generated from the following file:

- tracker/tracker_handlers.h

9.967 HETillumination::Trackermotion Struct Reference

Public Attributes

- float **dx2dtheta_itf**
- float **dx2dphi_itf**
- float **dy2dtheta_itf**
- float **dy2dphi_itf**
- float **drho2drho_itf**
- float **dx2dtel_x**
- float **dx2dtel_y**
- float **dy2dtel_x**
- float **dy2dtel_y**
- float **drho2dcarriage**
- float **drho2darm**

The documentation for this struct was generated from the following file:

- het_lib/illum_lib/include/illumination.h

9.968 TrackerPosition Class Reference

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)
- virtual void **apply_parms** (Payload &p) const

Public Attributes

- struct TrackerPosition::status **m_status**
- bool trajectory_

  flag indicating that TMCS is on a trajectory

- fptime t_

  latest reported index time from TMCS status

- itf_coordinate **itf_**
- rtf_coordinate **rtf_**
- tsf_coordinate **tsf_**
- bool known_

  flag indicating the position is known

The documentation for this class was generated from the following files:

- gui/include/trackerPositionData.h
- tcs/tracker_client.h
- gui/src/trackerPositionData.cpp
- tcs/tracker_client.cpp

9.969 TrackerStatus Class Reference

Classes

- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- struct TrackerStatus::status **m_status**

The documentation for this class was generated from the following files:

- gui/include/trackerStatusData.h
- gui/src/trackerStatusData.cpp

9.970 Ui::trackerStatusTab Class Reference

The documentation for this class was generated from the following file:

- gui/build/ui_trackerStatusTab.h

9.971 trackerStatusTabUi Class Reference

Public Slots

- void **UpdateTrackerStatus** (void)
- void **UpdateTrackerInitProgress** (int, bool, bool, bool, bool)
- void **TrackerInitComplete** (bool)
- void **UpdateTrackerErrors** (std::string, std::string)
- void **ClearTrackerErrors** (void)

Signals

- void **ClearTrackerErrors** (int)

Public Member Functions

- **trackerStatusTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)
- void **loadSettings** (void)
- void **saveSettings** (void)

The documentation for this class was generated from the following files:

- gui/include/trackerStatusTabUi.h
- gui/build/moc_trackerStatusTabUi.cpp
- gui/src/trackerStatusTabUi.cpp

9.972 liborchestrate::TrackerSubsystem Class Reference

Public Member Functions

- def **__init__**

The documentation for this class was generated from the following file:

- integration/liborchestrate.py

9.973 Ui::trackerTab Class Reference

The documentation for this class was generated from the following file:

- gui/build/ui_trackerTab.h

9.974 trackerTabUi Class Reference

Public Slots

- void **UpdateTrackerStatus** (void)
- void **UpdateTrackerPosition** (void)

Signals

- void **MoveTrackerITF** (std::string, std::string, double, double, double, double, double, double)
- void **MoveTrackerRTF** (std::string, std::string, double, double, double, double, double, double)
- void **MoveTrackerTSF** (std::string, std::string, double, double, double, double, double, double, double, double, double)
- void **MoveTrackerAng** (std::string, std::string, double, double)

Public Member Functions

- **trackerTabUi** (const char ∗name, QWidget ∗pLoadedWidget, guiConfig ∗pConfig)

Protected Member Functions

- string **setVelocity** (void)

The documentation for this class was generated from the following files:

- gui/include/trackerTabUi.h
- gui/build/moc_trackerTabUi.cpp
- gui/src/trackerTabUi.cpp

9.975 traj_class::Trajectory Class Reference

9.975.1 Detailed Description

```
This class contains everything we can possibly
know about one trajectory.
```

Public Member Functions

- def __init__
- def **someother_functions**

9.975.2 Member Function Documentation

9.975.2.1 def traj_class::Trajectory::__init__ (*self*, *traj_id* = `-1`, *db* = `None`)

```
Get all the information available for a trajectory from a
database and squirel it away for use by somebody.
```

The documentation for this class was generated from the following file:

- scripting/scripts/traj_class.py

9.976 Trajectory Class Reference

Public Member Functions

- **Trajectory** (unsigned id, Metrology &metrology, TrackerClient &tracker, Point p, TCSDriver &driver)
- bool **active** (void)
- void **thread_and_execute** (void)
- void **execute** (void)
- unsigned **id** (void)
- std::list< ShotPoint >::iterator **begin** ()
- std::list< ShotPoint >::iterator **end** ()
- const ShotPoint & **front** (void)
- bool **empty** (void)
- void **abort** (void)
- fptime **time_remaining** (void) const
- Message * **info** ()
- bool **align_to_now** ()
- void **apply_offset** (double dra, double ddec, double daz, double del, const pointing_coordinate &p, const weather &current_weather)
- void **correct_for_structure** (const pointing_coordinate &p, const weather &current_weather)
- double **initial_ra** (void)
- double **initial_dec** (void)
- void **apply_parms** (Payload &p) const
- void **offset_tiptilt** (const vec2_t &tp)
- void **notify** (const std::string &key, bool verbose=1)
- bool **attempted** (void)
- eq_coords **get_offset_register** (void)
- void **reset_offset_register** (const eq_coords &c)
- void **register_attempt** (void)
- void **increment_track** (void)

The documentation for this class was generated from the following files:

- tcs/trajectory.h
- tcs/trajectory.cpp

9.977 TrajectoryException Class Reference

Public Member Functions

- **TrajectoryException** (tcsexcept::tcs_exception_t __trajectory, const std::string &w, const char *file, int line, const char *func)

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_exception.h

9.978 traj_class::TrajectoryList Class Reference

9.978.1 Detailed Description

```
Do we need this class?
```

Public Member Functions

- def **__init__**

The documentation for this class was generated from the following file:

- scripting/scripts/traj_class.py

9.979 Ui::trajectoryTab Class Reference

The documentation for this class was generated from the following file:

- gui/build/ui_trajectoryTab.h

9.980 trajectoryTabUi Class Reference

Signals

- void **LoadRaDecTrajectory** (double, double, double, std::string, double, double)
- void **LoadXYTrajectory** (double, double, double)
- void **LoadFileTrajectory** (std::string, double, double)
- void **LoadSatelliteTrajectory** (std::string, std::string, double)
- void **LoadEphemerisTrajectory** (std::string, double, std::string)
- void **GoNext** (bool, bool, bool, bool)
- void **OffsetTrajectoryTipTilt** (double, double)
- void **OffsetTrajectoryRaDec** (double, double, bool)
- void **OffsetTrajectoryXY** (double, double, bool)
- void **OffsetTrajectoryRho** (double)
- void **OffsetFocus** (double)

Public Member Functions

- **trajectoryTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Protected Member Functions

- void **LoadRaDecTrajectory** (void)
- void **LoadXYTrajectory** (void)
- void **LoadFileTrajectory** (void)
- void **LoadSatelliteTrajectory** (void)
- void **LoadEphemerisTrajectory** (void)
- bool **BrowseForFile** (QString &file)

The documentation for this class was generated from the following files:

- gui/include/trajectoryTabUi.h
- gui/build/moc_trajectoryTabUi.cpp
- gui/src/trajectoryTabUi.cpp

9.981 tsf_coordinate Class Reference

9.981.1 Detailed Description

Class representing a point in Tracker Sender Frame coordinates.

`#include <tcs_coordinates.h>`

Public Member Functions

- **tsf_coordinate** (const fptime &t, const double &x, const double &y, const double &rho, const double &h1, const double &h2, const double &h3, const double &h4, const double &h5, const double &h6)
- virtual void **apply_parms** (Payload &p) const

Public Attributes

- fptime **t_**
- double **x_**
- double **ux_**
- double **lx_**
- double **y_**
- double **rho_**
- double **h1_**
- double **h2_**
- double **h3_**
- double **h4_**
- double **h5_**
- double **h6_**

The documentation for this class was generated from the following files:

- het_lib/common/include/tcs_coordinates.h
- het_lib/common/src/tcs_coordinates.cpp

9.982 tcs_photometry::TTCAMMetrologyData Struct Reference

Public Attributes

- Object **obj_**
- PhotData **pd_**

The documentation for this struct was generated from the following file:

- het_lib/photometry/include/object.h

9.983 tune< T > Class Template Reference

9.983.1 Detailed Description

template<typename T> class tune< T >

XXX

`#include <tcs_handler_doc.h>`

Public Member Functions

- **tune** (MetrologyLoop *l, MetrologyPIDLoop< T > *p, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.983.2 Member Function Documentation

9.983.2.1 template<typename T > void tune< T >::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/metrology.h

9.984 turn_off_leds Class Reference

9.984.1 Detailed Description

Turns off the LEDs on one or more controllers.

Optional parameters:

cntl<int> - Identifier of the controller to operate on, defaults to 0 for all.

`#include <camra_handler_doc.h>`

The documentation for this class was generated from the following file:

- camra/camra_handler_doc.h

9.985 camra_handlers::turn_off_leds Class Reference

Public Member Functions

- **turn_off_leds** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.985.1 Member Function Documentation

9.985.1.1 void camra_handlers::turn_off_leds::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- camra/camra_handlers.h

9.986 Ui_aboutDialog Class Reference

Public Member Functions

- void **setupUi** (QDialog *aboutDialog)
- void **retranslateUi** (QDialog *aboutDialog)

Public Attributes

- QLabel * **label**
- QWidget * **widget**
- QVBoxLayout * **verticalLayout_2**
- QScrollArea * **scrollArea**
- QWidget * **scrollAreaWidgetContents**
- QVBoxLayout * **verticalLayout**
- QLabel * **creditsLabel**
- QLabel * **buildInfoLabel**
- QLabel * **builtOnInfoLabel**
- QLabel * **builtAtInfoLabel**

The documentation for this class was generated from the following file:

- gui/build/ui_aboutDlg.h

9.987 Ui_acqCamStatusTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗acqCamStatusTab)
- void **retranslateUi** (QWidget ∗acqCamStatusTab)

Public Attributes

- QGroupBox ∗ **groupBox_24**
- QGridLayout ∗ **gridLayout_5**
- QGroupBox ∗ **groupBox**
- QGridLayout ∗ **gridLayout**
- QLedIndicator ∗ **acqFaultLedIndicator**
- QLedIndicator ∗ **acqWarningLedIndicator**
- QLedIndicator ∗ **acqTargetReachedLedIndicator**
- QLedIndicator ∗ **acqEnabledLedIndicator**
- QLedIndicator ∗ **acqLimitLedIndicator**
- QLedIndicator ∗ **acqFollowingErrorLedIndicator**
- QLedIndicator ∗ **acqSetPointLedIndicator**
- QLedIndicator ∗ **acqPowerLedIndicator**
- QLabel ∗ **label_107**
- QLabel ∗ **acqCameraFocusLabel**
- QLabel ∗ **label_108**
- QLabel ∗ **acqCameraMirrorLabel**
- QGroupBox ∗ **groupBox_25**
- QGridLayout ∗ **gridLayout_4**
- QLedIndicator ∗ **filterFaultLedIndicator**
- QLedIndicator ∗ **filterWarningLedIndicator**
- QLedIndicator ∗ **filterTargetReachedLedIndicator**
- QLedIndicator ∗ **filterEnabledLedIndicator**
- QLedIndicator ∗ **filterLimitLedIndicator**
- QLedIndicator ∗ **filterFollowingErrorLedIndicator**
- QLedIndicator ∗ **filterSetPointLedIndicator**
- QLabel ∗ **label_116**
- QLabel ∗ **filterPositionLabel**
- QSpacerItem ∗ **horizontalSpacer**
- QLabel ∗ **label**
- QGroupBox ∗ **groupBox_26**
- QGridLayout ∗ **gridLayout_7**
- QGroupBox ∗ **groupBox_27**
- QGridLayout ∗ **gridLayout_3**
- QLedIndicator ∗ **shutterFaultLedIndicator**
- QLedIndicator ∗ **shutterWarningLedIndicator**
- QLedIndicator ∗ **shutterTargetReachedLedIndicator**
- QLedIndicator ∗ **shutterEnabledLedIndicator**
- QLedIndicator ∗ **shutterLimitLedIndicator**
- QLedIndicator ∗ **shutterFollowingErrorLedIndicator**
- QLedIndicator ∗ **shutterSetPointLedIndicator**
- QLabel ∗ **label_131**

- QLabel * **shutterEncoderLabel**
- QLabel * **label_132**
- QLabel * **shutterPositionLabel**
- QGroupBox * **groupBox_28**
- QGridLayout * **gridLayout_6**
- QLedIndicator * **enwaArmFaultLedIndicator**
- QLedIndicator * **enwaArmWarningLedIndicator**
- QLedIndicator * **enwaArmTargetReachedLedIndicator**
- QLedIndicator * **enwaArmEnabledLedIndicator**
- QLedIndicator * **enwaArmLimitLedIndicator**
- QLedIndicator * **enwaArmFollowingErrorLedIndicator**
- QLedIndicator * **enwaArmSetPointLedIndicator**
- QLabel * **label_140**
- QLabel * **enwaEncoderLabel**
- QLabel * **label_141**
- QLabel * **enwaArmPositionLabel**
- QGroupBox * **groupBox_30**
- QGridLayout * **gridLayout_10**
- QGroupBox * **groupBox_2**
- QGridLayout * **gridLayout_9**
- QLedIndicator * **cwfsCameraPowerLedIndicator**
- QLabel * **label_111**
- QLabel * **cwfsLabel**
- QLabel * **label_110**
- QLabel * **cwfsCameraMirrorLabel**
- QGroupBox * **groupBox_31**
- QGridLayout * **gridLayout_8**
- QLedIndicator * **pvCameraPowerLedIndicator**
- QLabel * **label_2**
- QLabel * **label_112**
- QLabel * **pvLabel**
- QSpacerItem * **horizontalSpacer_4**
- QGroupBox * **groupBox_32**
- QGridLayout * **gridLayout_2**
- QLedIndicator * **sbc1PowerLedIndicator**
- QLedIndicator * **sw1PowerLedIndicator**
- QLabel * **label_136**
- QLabel * **ditherPositionLabel**
- QLedIndicator * **sbc2PowerLedIndicator**
- QLedIndicator * **sw2PowerLedIndicator**
- QSpacerItem * **horizontalSpacer_5**

The documentation for this class was generated from the following file:

- gui/build/ui_acqCamStatusTab.h

9.988 Ui_alertsDialog Class Reference

Public Member Functions

- void **setupUi** (QDialog *alertsDialog)
- void **retranslateUi** (QDialog *alertsDialog)

Public Attributes

- QGridLayout * **gridLayout_4**
- QTabWidget * **statusTabWidget**
- QWidget * **warningsTab**
- QGridLayout * **gridLayout_3**
- QSpacerItem * **horizontalSpacer**
- QPlainTextEdit * **warningsTextEdit**
- QCheckBox * **warningsClearOnExitCheckBox**
- QWidget * **errorsTabs**
- QGridLayout * **gridLayout_2**
- QPlainTextEdit * **errorsTextEdit**
- QSpacerItem * **horizontalSpacer_2**
- QCheckBox * **errorsClearOnExitCheckBox**
- QWidget * **fatalErrorsTab**
- QGridLayout * **gridLayout**
- QSpacerItem * **horizontalSpacer_3**
- QPlainTextEdit * **fatalErrorsTextEdit**
- QCheckBox * **fatalErrorsClearOnExitCheckBox**
- QWidget * **alarmTab**
- QGridLayout * **gridLayout_5**
- QSpacerItem * **horizontalSpacer_4**
- QPlainTextEdit * **alarmsTextEdit**
- QCheckBox * **alarmsClearOnExitCheckBox**
- QDialogButtonBox * **buttonBox**
- QCheckBox * **clearAllOnExitCheckBox**

The documentation for this class was generated from the following file:

- gui/build/ui_alertsDlg.h

9.989 Ui_apcTab Class Reference

Public Member Functions

- void **setupUi** (QMainWindow ∗apcTab)
- void **retranslateUi** (QMainWindow ∗apcTab)

Public Attributes

- QAction ∗ **viewCurrentAction**
- QAction ∗ **viewGroupsAction**
- QAction ∗ **viewMenuBarAction**
- QWidget ∗ **centralwidget**
- QGridLayout ∗ **apcTabGridLayout**
- QTcsFrame ∗ **apcControlTcsFrame**
- QGridLayout ∗ **apcControlTcsFrameGridLayout**
- QFrame ∗ **apcGroupsFrame**
- QGridLayout ∗ **apcGroupsFrameGridLayout**
- QTabWidget ∗ **apcControlTabWidget**
- QLabel ∗ **apcControlTcsFrameSpacingLabel**
- QMenuBar ∗ **apcTabMenuBar**
- QMenu ∗ **viewMenu**

The documentation for this class was generated from the following file:

- gui/build/ui_apcTab.h

9.990 Ui_cameraControlTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗cameraControlTab)
- void **retranslateUi** (QWidget ∗cameraControlTab)

Public Attributes

- QGridLayout ∗ **gridLayout_2**
- QTcsFrame ∗ **cameraControlTcsFrame**
- QGridLayout ∗ **gridLayout_3**
- QPushButton ∗ **setBinningPushButton**
- QCheckBox ∗ **storeImagesCheckBox**
- QLabel ∗ **label_19**
- QLabel ∗ **targetTempTextLabel**
- QLabel ∗ **targetTempLabel**
- QLineEdit ∗ **binningHeightLineEdit**
- QFrame ∗ **line**
- QLabel ∗ **label_3**
- QLabel ∗ **binningHeightLabel**
- QLineEdit ∗ **flushesLineEdit**
- QCheckBox ∗ **processImagesCheckBox**
- QPushButton ∗ **setIntervalSpeedPushButton**
- QLabel ∗ **label_9**
- QLabel ∗ **label_2**
- QCheckBox ∗ **cleanImagesCheckBox**
- QPushButton ∗ **setExposureTimePushButton**
- QLabel ∗ **label**
- QLabel ∗ **label_5**
- QLineEdit ∗ **targetTempLineEdit**
- QLabel ∗ **binningWidthLabel**
- QPushButton ∗ **setupCompletePushButton**
- QLabel ∗ **flushesLabel**
- QLabel ∗ **label_7**
- QLineEdit ∗ **intervalSpeedLineEdit**
- QLineEdit ∗ **coAddImagesLineEdit**
- QPushButton ∗ **setCoAddImagesPushButton**
- QLabel ∗ **exposureTimeLabel**
- QLabel ∗ **label_6**
- QPushButton ∗ **setFlushesPushButton**
- QLabel ∗ **cycleTimeLabel**
- QLineEdit ∗ **exposureTimeLineEdit**
- QLineEdit ∗ **binningWidthLineEdit**
- QLabel ∗ **label_4**
- QPushButton ∗ **setTargetTempPushButton**
- QLabel ∗ **coAddImagesLabel**
- QComboBox ∗ **viewExtensionComboBox**
- QLabel ∗ **label_10**

The documentation for this class was generated from the following file:

- gui/build/ui_cameraControlTab.h

9.991 Ui_cameraImageTab Class Reference

Public Member Functions

- void **setupUi** (QMainWindow ∗cameraImageTab)
- void **retranslateUi** (QMainWindow ∗cameraImageTab)

Public Attributes

- QAction ∗ **viewMarkersAction**
- QAction ∗ **clearGuideBoxAction**
- QAction ∗ **clearMeasuringTriangleAction**
- QAction ∗ **clearFromToMarkersAction**
- QAction ∗ **clearMarkersAction**
- QAction ∗ **clearManualCentroidAction**
- QAction ∗ **viewLocalPhotometryOptionsAction**
- QAction ∗ **viewSubApertureGraphicOverlayAction**
- QAction ∗ **viewPupilGraphicOverlayAction**
- QAction ∗ **viewObscurationGraphicOverlayAction**
- QAction ∗ **viewVectorsGraphicOverlayAction**
- QAction ∗ **viewM1GraphicOverlayAction**
- QAction ∗ **clearManualFiducialAction**
- QWidget ∗ **centralwidget**
- QGridLayout ∗ **gridLayout**
- QTcsFrame ∗ **cameraImageTcsFrame**
- QGridLayout ∗ **gridLayout_5**
- QLabel ∗ **label_2**
- QFrame ∗ **frame_3**
- QGridLayout ∗ **gridLayout_2**
- QLabel ∗ **label_14**
- QLabel ∗ **xPosLabel**
- QLabel ∗ **label_13**
- QLabel ∗ **targetTempLabel**
- QLabel ∗ **cameraTempLabel**
- QLabel ∗ **label_11**
- QLabel ∗ **yPosLabel**
- QLabel ∗ **label_12**
- QLabel ∗ **valueLabel**
- QSparkLine ∗ **cameraTempSparkLine**
- QSpacerItem ∗ **horizontalSpacer_4**
- QLabel ∗ **label**
- QSpacerItem ∗ **horizontalSpacer_10**
- QSpacerItem ∗ **horizontalSpacer_3**
- QGraphicsView ∗ **fullCameraImageGraphicsView**
- QGraphicsView ∗ **zoomAtCursorCameraImageGraphicsView**
- QGraphicsView ∗ **cameraImageGraphicsView**
- QFrame ∗ **frame**
- QGridLayout ∗ **gridLayout_4**
- QSpacerItem ∗ **horizontalSpacer_6**
- QLabel ∗ **label_10**

- QLabel * **frameCountLabel**
- QSpacerItem * **horizontalSpacer_2**
- QPushButton * **green_setGuideBoxPushButton**
- QPushButton * **yellow_clearGuideBoxPushButton**
- QPushButton * **yellow_clearMeasuringPushButton**
- QPushButton * **takeSingleImagePushButton**
- QPushButton * **green_setFiducialPushButton**
- QPushButton * **yellow_clearFiducialPushButton**
- QFrame * **frame_2**
- QGridLayout * **gridLayout_3**
- QLabel * **label_15**
- QSlider * **biasSlider**
- QLabel * **gainValueLabel**
- QSlider * **gainSlider**
- QLabel * **label_16**
- QLabel * **biasValueLabel**
- QMenuBar * **menubar**
- QMenu * **viewMenu**
- QMenu * **viewGraphicOverlays**
- QMenu * **clearMenu**

The documentation for this class was generated from the following file:

- gui/build/ui_cameraImageTab.h

9.992 Ui_cameraMetrologyControlTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗cameraMetrologyControlTab)
- void **retranslateUi** (QWidget ∗cameraMetrologyControlTab)

Public Attributes

- QTcsFrame ∗ **cameraMetrologyControlTcsFrame**
- QGridLayout ∗ **gridLayout**
- QLabel ∗ **label_5**
- QLabel ∗ **label_3**
- QLedIndicator ∗ **pipelineRunningLedIndicator**
- QPushButton ∗ **startStopPipelinePushButton**
- QPushButton ∗ **startStopLoopPushButton**
- QPushButton ∗ **startStopImagesPushButton**
- QPushButton ∗ **activateDeactivateLoopPushButton**
- QPushButton ∗ **resetMetrologyLoopPushButton**
- QLabel ∗ **label_13**
- QLineEdit ∗ **addDeltaXLineEdit**
- QLabel ∗ **getDeltaXLabel**
- QLabel ∗ **label_12**
- QLineEdit ∗ **addDeltaYLineEdit**
- QLabel ∗ **getDeltaYLabel**
- QLabel ∗ **label_9**
- QLineEdit ∗ **addDeltaZLineEdit**
- QLabel ∗ **getDeltaZLabel**
- QLabel ∗ **label_10**
- QLineEdit ∗ **addDeltaRhoLineEdit**
- QLabel ∗ **getDeltaRhoLabel**
- QLabel ∗ **label_11**
- QLineEdit ∗ **addDeltaThetaLineEdit**
- QLabel ∗ **getDeltaThetaLabel**
- QLabel ∗ **label_14**
- QLineEdit ∗ **addDeltaPhiLineEdit**
- QLabel ∗ **getDeltaPhiLabel**
- QPushButton ∗ **addCorrectionsPushButton**
- QPushButton ∗ **yellow_clearValuesPushButton**
- QPushButton ∗ **pasPhotometryOptionsPushButton**
- QGroupBox ∗ **groupBox**
- QGridLayout ∗ **gridLayout_2**
- QLabel ∗ **label_6**
- QComboBox ∗ **processorModesComboBox**
- QLabel ∗ **processingModeLabel**
- QLabel ∗ **label_4**
- QPushButton ∗ **centerProbePushButton**

The documentation for this class was generated from the following file:

- gui/build/ui_cameraMetrologyControlTab.h

9.993 Ui_cameraMetrologyDataTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗cameraMetrologyDataTab)
- void **retranslateUi** (QWidget ∗cameraMetrologyDataTab)

Public Attributes

- QGridLayout ∗ **gridLayout_20**
- QTcsFrame ∗ **cameraMetrologyDataTcsFrame**
- QGridLayout ∗ **gridLayout**
- QLabel ∗ **label_5**
- QLabel ∗ **centroidTextLabel**
- QLabel ∗ **centroidLabel**
- QLedIndicator ∗ **activeLedIndicator**
- QLedIndicator ∗ **validLedIndicator**
- QLedIndicator ∗ **ignoredLedIndicator**
- QLabel ∗ **fiducialTextLabel**
- QLabel ∗ **fiducialLabel**
- QLabel ∗ **label_10**
- QLabel ∗ **validityErrorLabel**
- QLabel ∗ **magnitudeAndXOffsetLabel**
- QSparkLine ∗ **magnitudeAndXOffsetSparkLine**
- QLabel ∗ **imageQualityAndYOffsetTextLabel**
- QLabel ∗ **imageQualityAndYOffsetLabel**
- QSparkLine ∗ **imageQualityAndYOffsetSparkLine**
- QLabel ∗ **transparencyAndFocusOffsetTextLabel**
- QLabel ∗ **transparencyAndFocusOffsetLabel**
- QSparkLine ∗ **transparencyAndFocusOffsetSparkLine**
- QLabel ∗ **peakFluxAndThetaOffsetTextLabel**
- QLabel ∗ **peakFluxAndThetaOffsetLabel**
- QSparkLine ∗ **peakFluxAndThetaOffsetSparkLine**
- QLabel ∗ **skyMagnitudeAndPhiOffsetTextLabel**
- QLabel ∗ **skyMagnitudeAndPhiOffsetLabel**
- QSparkLine ∗ **skyMagnitudeAndPhiOffsetSparkLine**
- QLabel ∗ **skyFluxAndGrocOffsetTextLabel**
- QLabel ∗ **skyFluxAndGrocOffsetLabel**
- QSparkLine ∗ **skyFluxAndGrocOffsetSparkLine**
- QLabel ∗ **magnitudeAndXOffsetTextLabel**

The documentation for this class was generated from the following file:

- gui/build/ui_cameraMetrologyDataTab.h

9.994 Ui_configureSystemsDialog Class Reference

Public Member Functions

- void **setupUi** (QDialog *configureSystemsDialog)
- void **retranslateUi** (QDialog *configureSystemsDialog)

Public Attributes

- QGridLayout * **gridLayout**
- QLabel * **label**
- QPlainTextEdit * **systemsTextEdit**
- QDialogButtonBox * **buttonBox**

The documentation for this class was generated from the following file:

- gui/build/ui_configureSystemsDlg.h

9.995 Ui_configureTabsDialog Class Reference

Public Member Functions

- void **setupUi** (QDialog *configureTabsDialog)
- void **retranslateUi** (QDialog *configureTabsDialog)

Public Attributes

- QGridLayout * **gridLayout**
- QDialogButtonBox * **buttonBox**
- QListWidget * **tabNamesListWidget**
- droppablePlainTextEdit * **newTabsPlainTextEdit**
- QLabel * **label**
- droppablePlainTextEdit * **currentTabsTextEdit**
- QLabel * **label_2**
- QLabel * **label_3**

The documentation for this class was generated from the following file:

- gui/build/ui_configureTabsDlg.h

9.996 Ui_confirmOffsetDialog Class Reference

Public Member Functions

- void **setupUi** (QDialog ∗confirmOffsetDialog)
- void **retranslateUi** (QDialog ∗confirmOffsetDialog)

Public Attributes

- QGridLayout ∗ **gridLayout**
- QRadioButton ∗ **moveTelescopeRadioButton**
- QRadioButton ∗ **moveProbeRadioButton**
- QRadioButton ∗ **moveTelescopeAndProbeRadioButton**
- QCheckBox ∗ **correctionCheckBox**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- gui/build/ui_confirmOffsetDlg.h

9.997 Ui_dmiTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗dmiTab)
- void **retranslateUi** (QWidget ∗dmiTab)

Public Attributes

- QGridLayout ∗ **gridLayout_2**
- QTcsFrame ∗ **dmiTcsFrame**
- QGridLayout ∗ **gridLayout**
- QLabel ∗ **label**
- QGroupBox ∗ **groupBox_23**
- QGridLayout ∗ **gridLayout_12**
- QLabel ∗ **label_2**
- QLedIndicator ∗ **dmiPipelineRunningLedIndicator**
- QPushButton ∗ **startStopDMILoopPushButton**
- QPushButton ∗ **activateDeactivateDMILoopPushButton**
- QLabel ∗ **label_3**
- QLineEdit ∗ **dmiCycleTimeLineEdit**
- QLineEdit ∗ **setDMIIntervalSpeedLineEdit**
- QPushButton ∗ **setDMICycleTimePushButton**
- QLabel ∗ **label_4**
- QLineEdit ∗ **focusLineEdit**
- QLineEdit ∗ **setFocusLineEdit**
- QPushButton ∗ **setFocusPushButton**
- QGroupBox ∗ **groupBox_26**
- QGridLayout ∗ **gridLayout_15**
- QLabel ∗ **label_43**
- QLabel ∗ **label_42**
- QLineEdit ∗ **sacDistanceLineEdit**
- QLineEdit ∗ **sacPeakAmplLineEdit**
- QLineEdit ∗ **sacCoupl1LineEdit**
- QLabel ∗ **label_41**
- QCheckBox ∗ **enableSACMeasurementsCheckBox**
- QLabel ∗ **label_46**
- QLineEdit ∗ **sacLastValidDistanceLineEdit**
- QGroupBox ∗ **groupBox_25**
- QGridLayout ∗ **gridLayout_14**
- QLabel ∗ **label_37**
- QLabel ∗ **label_39**
- QLabel ∗ **label_40**
- QLineEdit ∗ **ccasDistanceLineEdit**
- QLineEdit ∗ **ccasPeakAmplLineEdit**
- QLineEdit ∗ **ccasCoupl1LineEdit**
- QCheckBox ∗ **enableCCASMeasurementsCheckBox**
- QLabel ∗ **label_47**
- QLineEdit ∗ **ccasLastValidDistanceLineEdit**
- QPushButton ∗ **startStopDMIPipelinePushButton**

The documentation for this class was generated from the following file:

- gui/build/ui_dmiTab.h

9.998 Ui_domeTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗domeTab)
- void **retranslateUi** (QWidget ∗domeTab)

Public Attributes

- QGridLayout ∗ **gridLayout**
- QTcsFrame ∗ **domeTcsFrame**
- QGridLayout ∗ **gridLayout_3**
- QLabel ∗ **label**
- QGroupBox ∗ **groupBox_13**
- QGridLayout ∗ **gridLayout_2**
- QPushButton ∗ **homeDomePushButton**
- QPushButton ∗ **yellow_stopDomePushButton**
- QPushButton ∗ **ccasDomePushButton**
- QPushButton ∗ **red_abortDomePushButton**
- QPushButton ∗ **stowDomePushButton**
- QPushButton ∗ **red_resetDomePushButton**
- QLabel ∗ **label_11**
- QLineEdit ∗ **absDomeTargetLineEdit**
- QPushButton ∗ **green_moveDomeAbsPushButton**
- QLabel ∗ **label_21**
- QLineEdit ∗ **relDomeTargetLineEdit**
- QPushButton ∗ **green_moveDomeRelPushButton**
- QLabel ∗ **label_22**
- QComboBox ∗ **velocityComboBox**
- QPushButton ∗ **setVelocityPushButton**
- QGroupBox ∗ **groupBox**
- QGridLayout ∗ **gridLayout_5**
- QRadioButton ∗ **domeDirCCWRadioButton**
- QRadioButton ∗ **domeDirShortestRadioButton**
- QRadioButton ∗ **domeDirCWRadioButton**
- QSpacerItem ∗ **verticalSpacer_3**
- QGroupBox ∗ **groupBox_14**
- QGridLayout ∗ **gridLayout_7**
- QLineEdit ∗ **domeErrorNumLineEdit**
- QLabel ∗ **label_19**
- QBlinkingLineEdit ∗ **domePositionBlinkingLineEdit**
- QLabel ∗ **label_15**
- QLineEdit ∗ **domeTargetLineEdit**
- QLabel ∗ **label_12**
- QLineEdit ∗ **domeVelocityLineEdit**
- QLabel ∗ **label_23**
- QLineEdit ∗ **domeMaxVelocityLineEdit**
- QLabel ∗ **label_14**
- QLineEdit ∗ **domeDirectionLineEdit**
- QLabel ∗ **label_13**

- QLedIndicator * **domeMovingLedIndicator**
- QLabel * **label_18**
- QLedIndicator * **domePMACCommsLedIndicator**
- QLabel * **label_28**
- QLabel * **label_16**
- QLineEdit * **domeLocalRemoteLineEdit**
- QLabel * **label_17**
- QLedIndicator * **domeEMOLedIndicator**
- QLabel * **label_45**
- QLedIndicator * **domeStatusValidLedIndicator**
- QButtonGroup * **directionButtonGroup**

The documentation for this class was generated from the following file:

- gui/build/ui_domeTab.h

9.999 Ui_editPreferencesDialog Class Reference

Public Member Functions

- void **setupUi** (QDialog *editPreferencesDialog)
- void **retranslateUi** (QDialog *editPreferencesDialog)

Public Attributes

- QDialogButtonBox * **buttonBox**
- QTabWidget * **editPreferencesTabWidget**
- QWidget * **tab**
- QGroupBox * **groupBox**
- QGridLayout * **gridLayout**
- QRadioButton * **farenheitRadioButton**
- QRadioButton * **celciusRadioButton**
- QWidget * **tab_2**
- QGroupBox * **groupBox_2**
- QGridLayout * **gridLayout_2**
- QLabel * **label_2**
- QLabel * **label_5**
- QPushButton * **manualFiducialPushButton**
- QPushButton * **WFSSubApertureOverlayPushButton**
- QLabel * **label_15**
- QLabel * **label_14**
- QLabel * **label_13**
- QPushButton * **WFSPupilOverlayPushButton**
- QPushButton * **WFSObscurationOverlayPushButton**
- QPushButton * **WFSVectorsOverlayPushButton**
- QLabel * **label_6**
- QLabel * **label_8**
- QPushButton * **imageCursorPushButton**
- QPushButton * **guideBoxPushButton**
- QPushButton * **markerTextPushButton**
- QLabel * **label_9**
- QLabel * **label_11**
- QPushButton * **measuringTextPushButton**
- QPushButton * **measuringTrianglePushButton**
- QLabel * **label_12**
- QPushButton * **directionPushButton**
- QLabel * **label_10**
- QLabel * **label_16**
- QLabel * **label**
- QPushButton * **guiderCentroidFiducialPushButton**
- QLabel * **label_7**
- QPushButton * **manualCentroidBoxPushButton**
- QLabel * **label_3**
- QPushButton * **manualCentroidFiducialPushButton**
- QLabel * **label_4**
- QPushButton * **instrumentFiducialPushButton**

- QPushButton * **WFSM1OverlayPushButton**
- QPushButton * **WFSGridOverlayPushButton_2**

The documentation for this class was generated from the following file:

- gui/build/ui_editPreferencesDlg.h

9.1000 Ui_eventMonitorTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗eventMonitorTab)
- void **retranslateUi** (QWidget ∗eventMonitorTab)

Public Attributes

- QGridLayout ∗ **gridLayout_2**
- QTcsFrame ∗ **eventsTcsFrame**
- QGridLayout ∗ **gridLayout**
- QLabel ∗ **label_5**
- QLabel ∗ **label_6**
- QLabel ∗ **label**
- QLabel ∗ **label_2**
- QListWidget ∗ **keysListWidget**
- QPlainTextEdit ∗ **detailsPlainTextEdit**
- QPushButton ∗ **updatingPushButton**
- QPushButton ∗ **configureSystemsPushButton**
- QGroupBox ∗ **groupBox**
- QGridLayout ∗ **gridLayout_3**
- QLabel ∗ **label_7**
- QLabel ∗ **label_8**
- QLabel ∗ **label_9**
- QLineEdit ∗ **filterSystemLineEdit**
- QLineEdit ∗ **filterSourceLineEdit**
- QLineEdit ∗ **filterKeyLineEdit**
- QLabel ∗ **label_10**
- QPushButton ∗ **green_StartDBPushButton**
- QPushButton ∗ **red_StopDBPushButton**
- QPushButton ∗ **green_SetDBPathnamePushButton**
- QLineEdit ∗ **dbPathnameLineEdit**
- QPushButton ∗ **green_SetFilterPushButton**
- QPushButton ∗ **yellow_ResetFilterPushButton**
- QLabel ∗ **label_4**
- QLabel ∗ **numEventsLabel**
- QLabel ∗ **label_3**
- QLabel ∗ **maxEventsLabel**

The documentation for this class was generated from the following file:

- gui/build/ui_eventMonitorTab.h

9.1001 Ui_fcuControlTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗fcuControlTab)
- void **retranslateUi** (QWidget ∗fcuControlTab)

Public Attributes

- QTcsFrame ∗ **fcuControlTcsFrame**
- QGroupBox ∗ **groupBox**
- QGridLayout ∗ **gridLayout_3**
- QLabel ∗ **label_6**
- QLedIndicator ∗ **SliderALampPowerLedIndicator**
- QPushButton ∗ **greenSliderAExposePushButton**
- QLedIndicator ∗ **SliderALamp1PowerLedIndicator**
- QLedIndicator ∗ **SliderALamp4PowerLedIndicator**
- QPushButton ∗ **SliderAPositionPushButton**
- QLedIndicator ∗ **SliderALamp2PowerLedIndicator**
- QLedIndicator ∗ **SliderALamp3PowerLedIndicator**
- QLabel ∗ **SliderAPositionLabel**
- QComboBox ∗ **SliderAPositionComboBox**
- QLineEdit ∗ **SliderAExposureTimeLineEdit**
- QLabel ∗ **label_5**
- QPushButton ∗ **yellowSliderACancelExposurePushButton**
- QLabel ∗ **ShutterAStateLabel**
- QLabel ∗ **label_10**
- QLedIndicator ∗ **SliderALamp5PowerLedIndicator**
- QLabel ∗ **SliderAExposureTimeRemainingLabel**
- QGroupBox ∗ **groupBox_3**
- QGridLayout ∗ **gridLayout**
- QPushButton ∗ **yellowLDLSCancelExposurePushButton**
- QLabel ∗ **LDLSPositionLabel_2**
- QLabel ∗ **Label**
- QLabel ∗ **label_9**
- QLabel ∗ **label_4**
- QLineEdit ∗ **LDLSExposureTimeLineEdit**
- QPushButton ∗ **greenLDLSExposePushButton**
- QLabel ∗ **label_8**
- QLabel ∗ **LDLSShutterStateLabel**
- QLabel ∗ **label_13**
- QLedIndicator ∗ **LDLSPowerLedIndicator**
- QLabel ∗ **LDLSExposureTimeRemainingLabel**
- QGroupBox ∗ **groupBox_4**
- QGridLayout ∗ **gridLayout_4**
- QComboBox ∗ **InputLightBankPositionComboBox**
- QLabel ∗ **label_3**
- QComboBox ∗ **OutputLightGuidePositionComboBox**
- QLabel ∗ **InputLightBankPositionLabel**
- QLabel ∗ **FCUHeadPositionLabel**

- QLabel * **label_2**
- QPushButton * **DeployRetractFCUHeadPushButton**
- QLabel * **OutputLightGuidePositionLabel**
- QLabel * **label**
- QPushButton * **InputLightBankPositionPushButton**
- QPushButton * **OutputLightGuidePositionPushButton**
- QGroupBox * **groupBox_5**
- QGridLayout * **gridLayout_2**
- QPushButton * **SliderBPositionPushButton**
- QLedIndicator * **SliderBLamp1PowerLedIndicator**
- QLedIndicator * **SliderBLamp2PowerLedIndicator**
- QLabel * **SliderBPositionLabel**
- QComboBox * **SliderBPositionComboBox**
- QLedIndicator * **SliderBLamp3PowerLedIndicator**
- QLedIndicator * **SliderBLampPowerLedIndicator**
- QLabel * **label_11**
- QLineEdit * **SliderBExposureTimeLineEdit**
- QPushButton * **greenSliderBExposePushButton**
- QLedIndicator * **SliderBLamp4PowerLedIndicator**
- QLabel * **ShutterBStateLabel**
- QLabel * **label_7**
- QLedIndicator * **SliderBLamp5PowerLedIndicator**
- QLabel * **label_12**
- QPushButton * **yellowSliderBCancelExposurePushButton**
- QLabel * **SliderBExposureTimeRemainingLabel**
- QPushButton * **yellowAllLampsOffPushButton**

The documentation for this class was generated from the following file:

- gui/build/ui_fcuControlTab.h

9.1002 Ui_fcuStatusTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗fcuStatusTab)
- void **retranslateUi** (QWidget ∗fcuStatusTab)

Public Attributes

- QGridLayout ∗ **gridLayout_5**
- QGroupBox ∗ **groupBox_2**
- QGridLayout ∗ **gridLayout**
- QLabel ∗ **label_116**
- QLabel ∗ **slideraposLabel**
- QLabel ∗ **label_119**
- QLabel ∗ **lamp1atimeLabel**
- QLabel ∗ **lamp2atimeLabel**
- QLabel ∗ **lamp3atimeLabel**
- QLabel ∗ **lamp4atimeLabel**
- QLabel ∗ **lamp5atimeLabel**
- QLabel ∗ **label_117**
- QLabel ∗ **shutterastateLabel**
- QLabel ∗ **label_118**
- QLabel ∗ **lamp1awarmLabel**
- QLabel ∗ **lamp2awarmLabel**
- QLabel ∗ **lamp3awarmLabel**
- QLabel ∗ **lamp4awarmLabel**
- QLabel ∗ **lamp5awarmLabel**
- QLedIndicator ∗ **lamp1aLedIndicator**
- QLedIndicator ∗ **lamp2aLedIndicator**
- QLedIndicator ∗ **lamp3aLedIndicator**
- QLedIndicator ∗ **lamp4aLedIndicator**
- QLedIndicator ∗ **lamp5aLedIndicator**
- QGroupBox ∗ **groupBox_5**
- QGridLayout ∗ **gridLayout_4**
- QLabel ∗ **label_134**
- QLabel ∗ **sliderbposLabel**
- QLabel ∗ **label_135**
- QLabel ∗ **lamp1btimeLabel**
- QLabel ∗ **lamp2btimeLabel**
- QLabel ∗ **lamp3btimeLabel**
- QLabel ∗ **lamp4btimeLabel**
- QLabel ∗ **lamp5btimeLabel**
- QLabel ∗ **label_136**
- QLabel ∗ **shutterbstateLabel**
- QLabel ∗ **label_137**
- QLabel ∗ **lamp1bwarmLabel**
- QLabel ∗ **lamp2bwarmLabel**
- QLabel ∗ **lamp3bwarmLabel**
- QLabel ∗ **lamp4bwarmLabel**

- QLabel * **lamp5bwarmLabel**
- QLedIndicator * **lamp1bLedIndicator**
- QLedIndicator * **lamp2bLedIndicator**
- QLedIndicator * **lamp3bLedIndicator**
- QLedIndicator * **lamp4bLedIndicator**
- QLedIndicator * **lamp5bLedIndicator**
- QGroupBox * **groupBox_3**
- QGridLayout * **gridLayout_3**
- QSpacerItem * **horizontalSpacer**
- QLedIndicator * **ldlsLedIndicator**
- QLabel * **label_126**
- QLabel * **ldlswarmLabel**
- QSpacerItem * **horizontalSpacer_3**
- QLabel * **label_124**
- QLabel * **ldlsshutterstateLabel**
- QLabel * **label_125**
- QLabel * **ldlstimeLabel**
- QGroupBox * **groupBox_4**
- QGridLayout * **gridLayout_2**
- QLabel * **label_131**
- QLabel * **lightguideposLabel**
- QLabel * **fcuheadposLabel**
- QLabel * **label_130**
- QSpacerItem * **horizontalSpacer_4**
- QLabel * **label_138**
- QLabel * **inputbankposLabel**

The documentation for this class was generated from the following file:

- gui/build/ui_fcuStatusTab.h

9.1003 Ui_fiducialDialog Class Reference

Public Member Functions

- void **setupUi** (QDialog *fiducialDialog)
- void **retranslateUi** (QDialog *fiducialDialog)

Public Attributes

- QGridLayout * **gridLayout**
- QLabel * **label_2**
- QLabel * **label**
- QSpinBox * **ySpinBox**
- QSpinBox * **xSpinBox**
- QPushButton * **applyPushButton**

The documentation for this class was generated from the following file:

- gui/build/ui_fiducialDlg.h

9.1004 Ui_focalPlaneShutterTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗focalPlaneShutterTab)
- void **retranslateUi** (QWidget ∗focalPlaneShutterTab)

Public Attributes

- QGridLayout ∗ **gridLayout**
- QGraphicsView ∗ **focalPlaneShutterGraphicsView**
- QSpacerItem ∗ **horizontalSpacer**
- QLabel ∗ **label**
- QLabel ∗ **shutterPositionLabel**
- QPushButton ∗ **green_OpenShutterPushButton**
- QSpacerItem ∗ **horizontalSpacer_2**
- QLabel ∗ **label_2**
- QLabel ∗ **shutterEncoderAfterModuloLabel**
- QPushButton ∗ **yellow_CloseShutterPushButton**

The documentation for this class was generated from the following file:

- gui/build/ui_focalPlaneShutterTab.h

9.1005 Ui_guideBoxDialog Class Reference

Public Member Functions

- void **setupUi** (QDialog ∗guideBoxDialog)
- void **retranslateUi** (QDialog ∗guideBoxDialog)

Public Attributes

- QGridLayout ∗ **gridLayout**
- QLabel ∗ **label**
- QSpinBox ∗ **xSpinBox**
- QLabel ∗ **label_2**
- QSpinBox ∗ **ySpinBox**
- QLabel ∗ **label_3**
- QSpinBox ∗ **widthSpinBox**
- QLabel ∗ **label_4**
- QSpinBox ∗ **heightSpinBox**
- QPushButton ∗ **applyPushButton**

The documentation for this class was generated from the following file:

- gui/build/ui_guideBoxDlg.h

9.1006 Ui_iglooTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗iglooTab)
- void **retranslateUi** (QWidget ∗iglooTab)

Public Attributes

- QTcsFrame ∗ **tcsFrame**
- QWidget ∗ **verticalLayoutWidget_2**
- QVBoxLayout ∗ **verticalLayout_2**
- QLabel ∗ **label_spacer**
- QTabWidget ∗ **tabWidget**
- QWidget ∗ **tab_ctrl**
- QGroupBox ∗ **groupBox**
- QWidget ∗ **formLayoutWidget**
- QFormLayout ∗ **formLayout**
- QLabel ∗ **label**
- QLedIndicator ∗ **PowerEnableLedIndicator**
- QLabel ∗ **label_6**
- QLedIndicator ∗ **CoolantEnableLedIndicator**
- QGroupBox ∗ **groupBox_2**
- QWidget ∗ **verticalLayoutWidget**
- QVBoxLayout ∗ **verticalLayout**
- QCheckBox ∗ **CoolantShutoffDisableCheckBox**
- QCheckBox ∗ **ForcePowerOnCheckBox**
- QWidget ∗ **tab_status**
- QGridLayout ∗ **gridLayout_2**
- QVBoxLayout ∗ **StatusHeadingVBoxLayout**
- QScrollArea ∗ **StatusListScrollArea**
- QWidget ∗ **scrollAreaWidgetContents**
- QGridLayout ∗ **gridLayout**
- QVBoxLayout ∗ **StatusFieldsVBoxLayout**
- QGroupBox ∗ **groupBox_3**
- QWidget ∗ **formLayoutWidget_2**
- QFormLayout ∗ **formLayout_2**
- QLedIndicator ∗ **PLCConnectionLedIndicator**
- QLabel ∗ **label_2**
- QLabel ∗ **label_3**
- QLedIndicator ∗ **PLCConditionLedIndicator**

The documentation for this class was generated from the following file:

- gui/build/ui_iglooTab.h

9.1007 Ui_legacyTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗legacyTab)
- void **retranslateUi** (QWidget ∗legacyTab)

Public Attributes

- QGridLayout ∗ **gridLayout_13**
- QGroupBox ∗ **groupBox_20**
- QGridLayout ∗ **gridLayout_10**
- QGroupBox ∗ **groupBox_14**
- QGridLayout ∗ **gridLayout_7**
- QLineEdit ∗ **domeErrorNumLineEdit**
- QLabel ∗ **label_19**
- QBlinkingLineEdit ∗ **domePositionBlinkingLineEdit**
- QLabel ∗ **label_15**
- QLineEdit ∗ **domeTargetLineEdit**
- QLabel ∗ **label_12**
- QLineEdit ∗ **domeVelocityLineEdit**
- QLabel ∗ **label_23**
- QLineEdit ∗ **domeMaxVelocityLineEdit**
- QLabel ∗ **label_14**
- QLineEdit ∗ **domeDirectionLineEdit**
- QLabel ∗ **label_13**
- QLedIndicator ∗ **domeMovingLedIndicator**
- QLabel ∗ **label_18**
- QLedIndicator ∗ **domePMACCommsLedIndicator**
- QLabel ∗ **label_28**
- QLabel ∗ **label_16**
- QLineEdit ∗ **domeLocalRemoteLineEdit**
- QLabel ∗ **label_17**
- QLedIndicator ∗ **domeEMOLedIndicator**
- QLabel ∗ **label_45**
- QLedIndicator ∗ **domeStatusValidLedIndicator**
- QGroupBox ∗ **groupBox_13**
- QGridLayout ∗ **gridLayout_2**
- QPushButton ∗ **homeDomePushButton**
- QPushButton ∗ **yellow_stopDomePushButton**
- QPushButton ∗ **ccasDomePushButton**
- QPushButton ∗ **red_abortDomePushButton**
- QPushButton ∗ **stowDomePushButton**
- QPushButton ∗ **red_resetDomePushButton**
- QLabel ∗ **label_11**
- QLineEdit ∗ **absDomeTargetLineEdit**
- QPushButton ∗ **green_moveDomeAbsPushButton**
- QLabel ∗ **label_21**
- QLineEdit ∗ **relDomeTargetLineEdit**
- QPushButton ∗ **green_moveDomeRelPushButton**

- QLabel * **label_22**
- QComboBox * **velocityComboBox**
- QPushButton * **setVelocityPushButton**
- QGroupBox * **groupBox**
- QGridLayout * **gridLayout_5**
- QRadioButton * **domeDirShortestRadioButton**
- QRadioButton * **domeDirCWRadioButton**
- QRadioButton * **domeDirCCWRadioButton**
- QSpacerItem * **verticalSpacer_3**
- QGroupBox * **groupBox_21**
- QGridLayout * **gridLayout**
- QGroupBox * **groupBox_15**
- QGridLayout * **gridLayout_8**
- QPushButton * **yellow_stopShutterPushButton**
- QPushButton * **green_openShutterPushButton**
- QPushButton * **yellow_closeShutterPushButton**
- QPushButton * **red_abortShutterPushButton**
- QPushButton * **red_resetShutterPushButton**
- QGroupBox * **groupBox_16**
- QGridLayout * **gridLayout_3**
- QLabel * **label_31**
- QBlinkingLineEdit * **shutterPositionBlinkingLineEdit**
- QLabel * **label_34**
- QBlinkingLineEdit * **shutterMotionBlinkingLineEdit**
- QLabel * **label_24**
- QLineEdit * **shutterTargetLineEdit**
- QLabel * **label_29**
- QLedIndicator * **shutterMovingLedIndicator**
- QLabel * **label_30**
- QLedIndicator * **shutterPMACCommsLedIndicator**
- QLabel * **label_26**
- QLineEdit * **shutterErrorNumLineEdit**
- QLabel * **label_32**
- QLineEdit * **shutterLocalRemoteLineEdit**
- QLabel * **label_44**
- QLedIndicator * **shutterStatusValidLedIndicator**
- QSpacerItem * **verticalSpacer**
- QGroupBox * **groupBox_22**
- QGridLayout * **gridLayout_11**
- QGroupBox * **groupBox_17**
- QGridLayout * **gridLayout_9**
- QLabel * **label_20**
- QPushButton * **ccasStructurePushButton**
- QPushButton * **stowStructurePushButton**
- QPushButton * **yellow_stopStructurePushButton**
- QPushButton * **moveStructureToStructAzPushButton**
- QPushButton * **green_moveStructureToTrueAzPushButton**
- QLineEdit * **moveStructureTargetLineEdit**
- QGroupBox * **groupBox_18**
- QGridLayout * **gridLayout_6**

- QLedIndicator * **structureStatusValidLedIndicator**
- QLabel * **label_33**
- QBlinkingLineEdit * **structureAzimuthBlinkingLineEdit**
- QLabel * **label_27**
- QLabel * **label_25**
- QLedIndicator * **structureMovingLedIndicator**
- QLabel * **label_36**
- QLabel * **label_35**
- QLedIndicator * **structureInControlLedIndicator**
- QLineEdit * **structureTargetLineEdit**
- QGroupBox * **groupBox_19**
- QGridLayout * **gridLayout_4**
- QLabel * **label_38**
- QLineEdit * **structureFootHeightsLineEdit**
- QGroupBox * **groupBox_23**
- QGridLayout * **gridLayout_12**
- QLabel * **label_2**
- QLedIndicator * **dmiPipelineRunningLedIndicator**
- QPushButton * **startStopDMIILoopPushButton**
- QPushButton * **activateDeactivateDMIILoopPushButton**
- QLabel * **label**
- QLineEdit * **dmiCycleTimeLineEdit**
- QLineEdit * **setDMIIntervalSpeedLineEdit**
- QPushButton * **setDMICycleTimePushButton**
- QLabel * **label_3**
- QLineEdit * **focusLineEdit**
- QLineEdit * **setFocusLineEdit**
- QPushButton * **setFocusPushButton**
- QGroupBox * **groupBox_26**
- QGridLayout * **gridLayout_15**
- QLabel * **label_43**
- QLabel * **label_42**
- QLineEdit * **sacDistanceLineEdit**
- QLineEdit * **sacPeakAmplLineEdit**
- QLineEdit * **sacCoupl1LineEdit**
- QLabel * **label_41**
- QCheckBox * **enableSACMeasurementsCheckBox**
- QLabel * **label_46**
- QLineEdit * **sacLastValidDistanceLineEdit**
- QGroupBox * **groupBox_25**
- QGridLayout * **gridLayout_14**
- QLabel * **label_37**
- QLabel * **label_39**
- QLabel * **label_40**
- QLineEdit * **ccasDistanceLineEdit**
- QLineEdit * **ccasPeakAmplLineEdit**
- QLineEdit * **ccasCoupl1LineEdit**
- QCheckBox * **enableCCASMeasurementsCheckBox**
- QLabel * **label_47**
- QLineEdit * **ccasLastValidDistanceLineEdit**

- QPushButton * **startStopDMIPipelinePushButton**
- QSpacerItem * **horizontalSpacer**
- QSpacerItem * **verticalSpacer_2**
- QButtonGroup * **directionButtonGroup**

The documentation for this class was generated from the following file:

- gui/build/ui_legacyTab.h

9.1008 Ui_loadMarkersDialog Class Reference

Public Member Functions

- void **setupUi** (QDialog *loadMarkersDialog)
- void **retranslateUi** (QDialog *loadMarkersDialog)

Public Attributes

- QGridLayout * **gridLayout**
- QListWidget * **markersListWidget**
- QComboBox * **markerSetsComboBox**
- QLabel * **label_2**
- QDialogButtonBox * **buttonBox**
- QLabel * **label**

The documentation for this class was generated from the following file:

- gui/build/ui_loadMarkersDlg.h

9.1009 Ui_mainWindow Class Reference

Public Member Functions

- void **setupUi** (QMainWindow ∗mainWindow)
- void **retranslateUi** (QMainWindow ∗mainWindow)

Public Attributes

- QAction ∗ **actionFile_Quit**
- QAction ∗ **actionHelp_About**
- QAction ∗ **actionView_Dash_Board**
- QAction ∗ **actionFile_Preferences**
- QAction ∗ **actionWindow_Detach_Tab**
- QAction ∗ **actionWindow_Configure_Tabs**
- QAction ∗ **actionView_APC_Current**
- QAction ∗ **actionView_APC_Groups**
- QAction ∗ **actionView_Status_Bar**
- QAction ∗ **actionView_Menu_Bar**
- QAction ∗ **actionView_Tool_Bar**
- QAction ∗ **actionWhatsThis**
- QAction ∗ **actionHelp**
- QAction ∗ **actionView_Sound_Warnings**
- QAction ∗ **actionView_Sound_Alarms**
- QAction ∗ **actionView_Command_Status**
- QAction ∗ **actionView_Sound_FatalErrors**
- QAction ∗ **actionView_Sound_Errors**
- QWidget ∗ **centralwidget**
- QGridLayout ∗ **gridLayout_2**
- QTabWidget ∗ **mainGuiTabWidget**
- QMenuBar ∗ **mainWindowMenuBar**
- QMenu ∗ **menuFile**
- QMenu ∗ **menuHelp**
- QMenu ∗ **menuView**
- QMenu ∗ **actionView_Sound**
- QMenu ∗ **menuWindow**
- QStatusBar ∗ **mainWindowStatusBar**
- QDockWidget ∗ **dashBoardDockWidget**
- QWidget ∗ **dockWidgetContents**
- QGridLayout ∗ **gridLayout**
- QPushButton ∗ **UTCPushButton**
- QPushButton ∗ **RAPushButton**
- QPushButton ∗ **structAZPushButton**
- QPushButton ∗ **trackTimeUntilStartPushButton**
- QPushButton ∗ **RSEPushButton**
- QPushButton ∗ **red_abortAllPushButton**
- QPushButton ∗ **LSTPushButton**
- QPushButton ∗ **DECPushButton**
- QPushButton ∗ **domeAZPushButton**
- QPushButton ∗ **trackTimeRemainingPushButton**

- QPushButton * **targetRSEPushButton**
- QPushButton * **TCSAlertsPushButton**
- QPushButton * **PASAlertsPushButton**
- QPushButton * **LegacyAlertsPushButton**
- QPushButton * **DomeAlertsPushButton**
- QPushButton * **WeatherAlertsPushButton**
- QPushButton * **VIRUSAlertsPushButton**
- QPushButton * **APCAlertsPushButton**
- QPushButton * **LogRelayAlertsPushButton**
- QPushButton * **OCDAlertsPushButton**
- QPushButton * **IglooAlertsPushButton**
- QPushButton * **HRSAlertsPushButton**
- QPushButton * **TrackerAlertsPushButton**
- QPushButton * **PFIPAlertsPushButton**
- QPushButton * **StructureAlertsPushButton**
- QPushButton * **ShutterAlertsPushButton**
- QPushButton * **SAMSAlertsPushButton**
- QPushButton * **LRS2AlertsPushButton**
- QPushButton * **NamedAlertsPushButton**
- QPushButton * **EventMonitorAlertsPushButton**
- QPushButton * **CCASAlertsPushButton**
- QPushButton * **HPFAlertsPushButton**
- QPushButton * **SpareAlertsPushButton**
- QToolBar * **mainWindowToolBar**
- QDockWidget * **commandStatusDockWidget**
- QWidget * **dockWidgetContents_2**
- QGridLayout * **gridLayout_3**
- QTextEdit * **commandStatusTextEdit**

The documentation for this class was generated from the following file:

- gui/build/ui_mainWindow.h

9.1010 Ui_pfipControlTab Class Reference

Public Member Functions

- void **setupUi** (QWidget *pfipControlTab)
- void **retranslateUi** (QWidget *pfipControlTab)

Public Attributes

- QTcsFrame * **pfipControlTcsFrame**
- QGridLayout * **gridLayout**
- QLabel * **DitherPositionLabel**
- QComboBox * **DitherSelectionComboBox**
- QLabel * **ACQCameraMirrorPositionLabel**
- QLabel * **label_9**
- QPushButton * **selectPV_CWFSCameraPushButton**
- QLedIndicator * **PVCameraPowerLedIndicator**
- QLabel * **label_5**
- QLabel * **label_2**
- QLabel * **ACQCameraFocusPositionLabel**
- QLineEdit * **ACQCameraFocusLineEdit**
- QPushButton * **setACQCameraFocusAbsPushButton**
- QLedIndicator * **SBC1_3PowerLedIndicator**
- QLabel * **PV_CWFSCameraMirrorPositionLabel**
- QLedIndicator * **TTCameraPowerLedIndicator**
- QLedIndicator * **Guider2CameraPowerLedIndicator**
- QLabel * **label_15**
- QLabel * **Guider2FilterPositionLabel**
- QLedIndicator * **Guider1CameraPowerLedIndicator**
- QLabel * **label_14**
- QLabel * **Guider1FilterPositionLabel**
- QPushButton * **homeShutterPushButton**
- QPushButton * **powerOnAllPushButton**
- QPushButton * **powerOffAllPushButton**
- QPushButton * **deployRetractPV_CWFSCameraMirrorPushButton**
- QLedIndicator * **EthernetSwitch1PowerLedIndicator**
- QLabel * **label_16**
- QLabel * **ShutterPositionLabel**
- QLedIndicator * **ACQCameraPowerLedIndicator**
- QPushButton * **deployRetractACQCameraMirrorPushButton**
- QLabel * **label_6**
- QLabel * **label_4**
- QLabel * **PV_CWFSCameraPositionLabel**
- QLedIndicator * **WFS1CameraPowerLedIndicator**
- QLedIndicator * **WFS2CameraPowerLedIndicator**
- QLabel * **label_7**
- QLabel * **label_1**
- QLabel * **ACQCameraFilterPositionLabel**
- QComboBox * **Guider2FilterSelectionComboBox**
- QPushButton * **setGuider2FilterPushButton**

- QLabel * **label_17**
- QLabel * **ShutterEncoderAfterModuloLabel**
- QLabel * **label_8**
- QComboBox * **Guider1FilterSelectionComboBox**
- QPushButton * **setGuider1FilterPushButton**
- QPushButton * **setDitherPushButton**
- QLedIndicator * **BIBCameraPowerLedIndicator**
- QLedIndicator * **HPF_ACQCameraPowerLedIndicator**
- QPushButton * **green_OpenShutterPushButton**
- QPushButton * **yellow_CloseShutterPushButton**
- QLedIndicator * **HR_SASCameraPowerLedIndicator**
- QLabel * **label_3**
- QLabel * **label_10**
- QPushButton * **setACQCameraFocusRelPushButton**
- QLedIndicator * **SBC4_5PowerLedIndicator**
- QLedIndicator * **CWFSCameraPowerLedIndicator**
- QLabel * **label**
- QPushButton * **setACQCameraFilterPushButton**
- QLedIndicator * **EthernetSwitch2PowerLedIndicator**
- QComboBox * **ACQCameraFilterSelectionComboBox**

The documentation for this class was generated from the following file:

- gui/build/ui_pfipControlTab.h

9.1011 Ui_pfipEnvironmentTab Class Reference

Public Member Functions

- void **setupUi** (QWidget *pfipEnvironmentTab)
- void **retranslateUi** (QWidget *pfipEnvironmentTab)

Public Attributes

- QTcsFrame * **tcsFrame**
- QLabel * **label**
- QGroupBox * **groupBox_24**
- QGridLayout * **gridLayout**
- QLabel * **label_107**
- QLabel * **label_109**
- QLedIndicator * **UpperTempAlarmLedIndicator**
- QLabel * **LowerTempLabel**
- QLedIndicator * **UpperFluidAlarmLedIndicator**
- QSpacerItem * **horizontalSpacer**
- QLabel * **label_121**
- QSparkLine * **LowerTempSparkLine**
- QLedIndicator * **LowerFluidAlarmLedIndicator**
- QLabel * **label_120**
- QSparkLine * **UpperTempSparkLine**
- QLabel * **UpperTempLabel**
- QSparkLine * **DomeTempSparkLine**
- QLabel * **label_114**
- QLabel * **DomeTempLabel**
- QLedIndicator * **LowerTempAlarmLedIndicator**
- QGroupBox * **groupBox**
- QGridLayout * **gridLayout_3**
- QSparkLine * **FPAPurgeAirDewPointTempSparkLine**
- QLabel * **FPAPurgeAirTempLabel**
- QLabel * **label_117**
- QSparkLine * **FPAGlycolOutTempSparkLine**
- QLabel * **label_119**
- QLabel * **label_111**
- QSparkLine * **FPATempSparkLine**
- QLabel * **label_115**
- QLabel * **FPAPurgeAirFlowLabel**
- QLabel * **FPAPurgeAirDewPointTempLabel**
- QSparkLine * **FPAGlycolInTempSparkLine**
- QSparkLine * **FPAPurgeAirTempSparkLine**
- QLabel * **FPATempLabel**
- QSparkLine * **FPAPurgeAirFlowSparkLine**
- QLabel * **label_116**
- QLabel * **FPAGlycolInTempLabel**
- QLabel * **label_118**
- QLabel * **FPAGlycolOutTempLabel**
- QGroupBox * **groupBox_2**

- QGridLayout * **gridLayout_4**
- QLabel * **label_124**
- QLabel * **WFCPurgeGN2FlowLabel**
- QLabel * **WFCPurgeGN2DewPointTempLabel**
- QLabel * **label_126**
- QLabel * **label_112**
- QSparkLine * **WFCTempSparkLine**
- QLabel * **WFCPurgeGN2TempLabel**
- QSparkLine * **WFCPurgeGN2TempSparkLine**
- QSparkLine * **WFCPurgeGN2FlowSparkLine**
- QSparkLine * **WFCPurgeGN2DewPointTempSparkLine**
- QLabel * **WFCTempLabel**
- QLabel * **label_122**

The documentation for this class was generated from the following file:

- gui/build/ui_pfipEnvironmentTab.h

9.1012 Ui_photometryOptionsDialog Class Reference

Public Member Functions

- void **setupUi** (QDialog ∗photometryOptionsDialog)
- void **retranslateUi** (QDialog ∗photometryOptionsDialog)

Public Attributes

- QGridLayout ∗ **gridLayout**
- QLabel ∗ **label**
- QLineEdit ∗ **magZeroLineEdit**
- QLabel ∗ **label_10**
- QLineEdit ∗ **saturationLineEdit**
- QLabel ∗ **label_5**
- QLineEdit ∗ **detectSigmaLineEdit**
- QLabel ∗ **label_4**
- QLineEdit ∗ **skyOrderLineEdit**
- QLabel ∗ **label_6**
- QLineEdit ∗ **minPixelsLineEdit**
- QLabel ∗ **label_11**
- QLineEdit ∗ **skyAnnulusLineEdit**
- QLabel ∗ **label_7**
- QLineEdit ∗ **maxPixelsLineEdit**
- QLabel ∗ **label_3**
- QLineEdit ∗ **kronApertureLineEdit**
- QLabel ∗ **label_8**
- QLineEdit ∗ **gainLineEdit**
- QLabel ∗ **label_2**
- QLineEdit ∗ **fixedApertureLineEdit**
- QLabel ∗ **label_9**
- QLineEdit ∗ **nanValueLineEdit**
- QPushButton ∗ **applyPushButton**

The documentation for this class was generated from the following file:

- gui/build/ui_photometryOptionsDlg.h

9.1013 Ui_probeAssemblyTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗probeAssemblyTab)
- void **retranslateUi** (QWidget ∗probeAssemblyTab)

Public Attributes

- QGridLayout ∗ **gridLayout**
- QGraphicsView ∗ **probeAssemblyGraphicsView**
- QCheckBox ∗ **guider1ShowRangeCheckBox**
- QCheckBox ∗ **wfs1ShowRangeCheckBox**
- QCheckBox ∗ **guider2ShowRangeCheckBox**
- QCheckBox ∗ **wfs2ShowRangeCheckBox**

The documentation for this class was generated from the following file:

- gui/build/ui_probeAssemblyTab.h

9.1014 Ui_probeControlTab Class Reference

Public Member Functions

- void **setupUi** (QWidget *probeControlTab)
- void **retranslateUi** (QWidget *probeControlTab)

Public Attributes

- QTcsFrame * **probeControlTcsFrame**
- QPushButton * **green_beginRangeProbesPushButton**
- QPushButton * **red_disableProbesPushButton**
- QLabel * **label_6**
- QPushButton * **green_moveProbesPushButton**
- QGroupBox * **groupBox_8**
- QPushButton * **yellow_guider1ClearPushButton**
- QGroupBox * **groupBox**
- QGridLayout * **gridLayout_2**
- QLabel * **label_14**
- QLabel * **guider1ProbeAngleLabel**
- QLabel * **label**
- QLabel * **guider1ProbePositionLabel**
- QLabel * **label_21**
- QLabel * **guider1ProbeEncoderLabel**
- QLineEdit * **guider1ProbePositionLineEdit**
- QGroupBox * **groupBox_2**
- QGridLayout * **gridLayout_5**
- QLabel * **label_20**
- QLabel * **guider1ArmAngleLabel**
- QLabel * **label_3**
- QLabel * **guider1ArmPositionLabel**
- QLabel * **label_22**
- QLabel * **guider1ArmEncoderLabel**
- QLineEdit * **guider1ArmPositionLineEdit**
- QGroupBox * **groupBox_3**
- QGridLayout * **gridLayout_6**
- QLedIndicator * **guider1ProbeLimitLedIndicator**
- QLedIndicator * **guider1ArmSetPointAckLedIndicator**
- QLabel * **label_33**
- QLedIndicator * **guider1ArmLimitLedIndicator**
- QLedIndicator * **guider1ProbeEnabledLedIndicator**
- QLedIndicator * **guider1ProbeTargetReachedLedIndicator**
- QLedIndicator * **guider1ArmEnabledLedIndicator**
- QLedIndicator * **guider1ArmTargetReachedLedIndicator**
- QLedIndicator * **guider1ProbeSetPointAckLedIndicator**
- QLabel * **label_11**
- QLedIndicator * **guider1ProbeFollowingErrorLedIndicator**
- QLedIndicator * **guider1ArmFollowingErrorLedIndicator**
- QLedIndicator * **guider1ProbeWarningLedIndicator**
- QLedIndicator * **guider1ArmWarningLedIndicator**

- QLedIndicator * **guider1ProbeFaultLedIndicator**
- QLedIndicator * **guider1ArmFaultLedIndicator**
- QPushButton * **green_endRangeProbesPushButton**
- QPushButton * **yellow_clearAllPushButton**
- QPushButton * **green_homeProbesPushButton**
- QPushButton * **red_interruptPushButton**
- QGroupBox * **groupBox_9**
- QPushButton * **yellow_guider2ClearPushButton**
- QGroupBox * **groupBox_4**
- QGridLayout * **gridLayout_3**
- QLabel * **label_15**
- QLabel * **guider2ProbeAngleLabel**
- QLabel * **label_2**
- QLabel * **guider2ProbePositionLabel**
- QLabel * **label_23**
- QLabel * **guider2ProbeEncoderLabel**
- QLineEdit * **guider2ProbePositionLineEdit**
- QGroupBox * **groupBox_5**
- QGridLayout * **gridLayout_7**
- QLabel * **label_24**
- QLabel * **guider2ArmAngleLabel**
- QLabel * **label_4**
- QLabel * **guider2ArmPositionLabel**
- QLabel * **label_25**
- QLabel * **guider2ArmEncoderLabel**
- QLineEdit * **guider2ArmPositionLineEdit**
- QGroupBox * **groupBox_6**
- QGridLayout * **gridLayout_8**
- QLedIndicator * **guider2ProbeLimitLedIndicator**
- QLedIndicator * **guider2ArmSetPointAckLedIndicator**
- QLabel * **label_34**
- QLedIndicator * **guider2ArmLimitLedIndicator**
- QLedIndicator * **guider2ProbeEnabledLedIndicator**
- QLedIndicator * **guider2ProbeTargetReachedLedIndicator**
- QLedIndicator * **guider2ArmEnabledLedIndicator**
- QLedIndicator * **guider2ArmTargetReachedLedIndicator**
- QLedIndicator * **guider2ProbeSetPointAckLedIndicator**
- QLabel * **label_12**
- QLedIndicator * **guider2ProbeFollowingErrorLedIndicator**
- QLedIndicator * **guider2ArmFollowingErrorLedIndicator**
- QLedIndicator * **guider2ProbeWarningLedIndicator**
- QLedIndicator * **guider2ArmWarningLedIndicator**
- QLedIndicator * **guider2ProbeFaultLedIndicator**
- QLedIndicator * **guider2ArmFaultLedIndicator**
- QGroupBox * **groupBox_13**
- QPushButton * **yellow_wfs1ClearPushButton**
- QGroupBox * **groupBox_14**
- QGridLayout * **gridLayout_11**
- QLabel * **label_17**
- QLabel * **wfs1ProbeAngleLabel**

- QLabel * **label_8**
- QLabel * **wfs1ProbePositionLabel**
- QLabel * **label_29**
- QLabel * **wfs1ProbeEncoderLabel**
- QLineEdit * **wfs1ProbePositionLineEdit**
- QGroupBox * **groupBox_15**
- QGridLayout * **gridLayout_12**
- QLabel * **label_30**
- QLabel * **wfs1ArmAngleLabel**
- QLabel * **label_9**
- QLabel * **wfs1ArmPositionLabel**
- QLabel * **label_31**
- QLabel * **wfs1ArmEncoderLabel**
- QLineEdit * **wfs1ArmPositionLineEdit**
- QGroupBox * **groupBox_16**
- QGridLayout * **gridLayout_13**
- QLedIndicator * **wfs1ProbeLimitLedIndicator**
- QLedIndicator * **wfs1ArmSetPointAckLedIndicator**
- QLabel * **label_36**
- QLedIndicator * **wfs1ArmLimitLedIndicator**
- QLedIndicator * **wfs1ProbeEnabledLedIndicator**
- QLedIndicator * **wfs1ProbeTargetReachedLedIndicator**
- QLedIndicator * **wfs1ArmEnabledLedIndicator**
- QLedIndicator * **wfs1ArmTargetReachedLedIndicator**
- QLedIndicator * **wfs1ProbeSetPointAckLedIndicator**
- QLabel * **label_18**
- QLedIndicator * **wfs1ProbeFollowingErrorLedIndicator**
- QLedIndicator * **wfs1ArmFollowingErrorLedIndicator**
- QLedIndicator * **wfs1ProbeWarningLedIndicator**
- QLedIndicator * **wfs1ArmWarningLedIndicator**
- QLedIndicator * **wfs1ProbeFaultLedIndicator**
- QLedIndicator * **wfs1ArmFaultLedIndicator**
- QGroupBox * **groupBox_17**
- QPushButton * **yellow_wfs2ClearPushButton**
- QGroupBox * **groupBox_18**
- QGridLayout * **gridLayout_14**
- QLabel * **label_19**
- QLabel * **wfs2ProbeAngleLabel**
- QLabel * **label_10**
- QLabel * **wfs2ProbePositionLabel**
- QLabel * **label_32**
- QLabel * **wfs2ProbeEncoderLabel**
- QLineEdit * **wfs2ProbePositionLineEdit**
- QGroupBox * **groupBox_19**
- QGridLayout * **gridLayout_15**
- QLabel * **label_37**
- QLabel * **wfs2ArmAngleLabel**
- QLabel * **label_38**
- QLabel * **wfs2ArmPositionLabel**
- QLabel * **label_39**

- QLabel * **wfs2ArmEncoderLabel**
- QLineEdit * **wfs2ArmPositionLineEdit**
- QGroupBox * **groupBox_20**
- QGridLayout * **gridLayout_16**
- QLedIndicator * **wfs2ProbeLimitLedIndicator**
- QLedIndicator * **wfs2ArmSetPointAckLedIndicator**
- QLabel * **label_40**
- QLedIndicator * **wfs2ArmLimitLedIndicator**
- QLedIndicator * **wfs2ProbeEnabledLedIndicator**
- QLedIndicator * **wfs2ProbeTargetReachedLedIndicator**
- QLedIndicator * **wfs2ArmEnabledLedIndicator**
- QLedIndicator * **wfs2ArmTargetReachedLedIndicator**
- QLedIndicator * **wfs2ProbeSetPointAckLedIndicator**
- QLabel * **label_41**
- QLedIndicator * **wfs2ProbeFollowingErrorLedIndicator**
- QLedIndicator * **wfs2ArmFollowingErrorLedIndicator**
- QLedIndicator * **wfs2ProbeWarningLedIndicator**
- QLedIndicator * **wfs2ArmWarningLedIndicator**
- QLedIndicator * **wfs2ProbeFaultLedIndicator**
- QLedIndicator * **wfs2ArmFaultLedIndicator**

The documentation for this class was generated from the following file:

- gui/build/ui_probeControlTab.h

9.1015 Ui_probeGainControlTab Class Reference

Public Member Functions

- void **setupUi** (QWidget *probeGainControlTab)
- void **retranslateUi** (QWidget *probeGainControlTab)

Public Attributes

- QTcsFrame * **tcsFrame_8**
- QGridLayout * **gridLayout**
- QPushButton * **green_setGainsPushButton**
- QLineEdit * **wfs1IGainLineEdit**
- QLabel * **label_2**
- QPushButton * **clearGainsPushButton**
- QLineEdit * **wfs2PGainLineEdit**
- QLineEdit * **guider2PGainLineEdit**
- QLabel * **label**
- QLabel * **label_255**
- QLineEdit * **wfs2IGainLineEdit**
- QLineEdit * **wfs1PGainLineEdit**
- QLineEdit * **guider1IGainLineEdit**
- QLineEdit * **guider1PGainLineEdit**
- QLineEdit * **guider2IGainLineEdit**
- QLabel * **label_3**
- QLabel * **label_6**
- QLabel * **label_5**
- QLabel * **label_4**

The documentation for this class was generated from the following file:

- gui/build/ui_probeGainControlTab.h

9.1016 Ui_probeStatusTab Class Reference

Public Member Functions

- void **setupUi** (QWidget *probeStatusTab)
- void **retranslateUi** (QWidget *probeStatusTab)

Public Attributes

- QGroupBox * **groupBox_2**
- QGridLayout * **gridLayout**
- QLedIndicator * **guider1PowerLedIndicator**
- QLabel * **label**
- QLabel * **guider1ProbePositionLabel**
- QLabel * **label_2**
- QLabel * **guider1ArmPositionLabel**
- QLedIndicator * **guider1ConfiguredLedIndicator**
- QGroupBox * **groupBox_4**
- QLedIndicator * **guider1ArmFaultLedIndicator**
- QLedIndicator * **guider1ArmWarningLedIndicator**
- QLedIndicator * **guider1ArmTargetReachedLedIndicator**
- QLedIndicator * **guider1ArmEnabledLedIndicator**
- QLedIndicator * **guider1ArmLimitLedIndicator**
- QLedIndicator * **guider1ArmFollowingErrorLedIndicator**
- QLedIndicator * **guider1ArmSetPointLedIndicator**
- QLabel * **label_44**
- QLabel * **guider1ArmEncoderYLabel**
- QLabel * **label_45**
- QLabel * **guider1ArmAngleLabel**
- QGroupBox * **groupBox_3**
- QLedIndicator * **guider1ProbeFaultLedIndicator**
- QLedIndicator * **guider1ProbeWarningLedIndicator**
- QLedIndicator * **guider1ProbeTargetReachedLedIndicator**
- QLedIndicator * **guider1ProbeEnabledLedIndicator**
- QLedIndicator * **guider1ProbeLimitLedIndicator**
- QLedIndicator * **guider1ProbeFollowingErrorLedIndicator**
- QLedIndicator * **guider1ProbeSetPointLedIndicator**
- QLabel * **label_28**
- QLabel * **guider1ProbeEncoderXLabel**
- QLabel * **label_34**
- QLabel * **guider1ProbeAngleLabel**
- QGroupBox * **groupBox_13**
- QLedIndicator * **guider1FilterFaultLedIndicator**
- QLedIndicator * **guider1FilterWarningLedIndicator**
- QLedIndicator * **guider1FilterTargetReachedLedIndicator**
- QLedIndicator * **guider1FilterEnabledLedIndicator**
- QLedIndicator * **guider1FilterLimitLedIndicator**
- QLedIndicator * **guider1FilterFollowingErrorLedIndicator**
- QLedIndicator * **guider1FilterSetPointLedIndicator**
- QLabel * **label_29**

- QLabel * **guider1FilterLabel**
- QGroupBox * **groupBox_9**
- QGridLayout * **gridLayout_11**
- QLedIndicator * **guider2PowerLedIndicator**
- QLabel * **label_4**
- QLabel * **guider2ProbePositionLabel**
- QLabel * **label_3**
- QLabel * **guider2ArmPositionLabel**
- QLedIndicator * **guider2ConfiguredLedIndicator**
- QGroupBox * **groupBox_11**
- QGridLayout * **gridLayout_3**
- QLedIndicator * **guider2ArmFaultLedIndicator**
- QLedIndicator * **guider2ArmWarningLedIndicator**
- QLedIndicator * **guider2ArmLimitLedIndicator**
- QLedIndicator * **guider2ArmEnabledLedIndicator**
- QLedIndicator * **guider2ArmSetPointLedIndicator**
- QLabel * **guider2ArmEncoderYLabel**
- QLabel * **label_62**
- QLedIndicator * **guider2ArmTargetReachedLedIndicator**
- QLedIndicator * **guider2ArmFollowingErrorLedIndicator**
- QLabel * **guider2ArmAngleLabel**
- QLabel * **label_63**
- QGroupBox * **groupBox_10**
- QGridLayout * **gridLayout_9**
- QLedIndicator * **guider2ProbeFaultLedIndicator**
- QLedIndicator * **guider2ProbeWarningLedIndicator**
- QLedIndicator * **guider2ProbeTargetReachedLedIndicator**
- QLedIndicator * **guider2ProbeEnabledLedIndicator**
- QLedIndicator * **guider2ProbeLimitLedIndicator**
- QLedIndicator * **guider2ProbeFollowingErrorLedIndicator**
- QLedIndicator * **guider2ProbeSetPointLedIndicator**
- QLabel * **guider2ProbeEncoderX**
- QLabel * **guider2ProbeEncoderXLabel**
- QLabel * **guider2ProbeAngle**
- QLabel * **guider2ProbeAngleLabel**
- QGroupBox * **groupBox_24**
- QGridLayout * **gridLayout_10**
- QLedIndicator * **guider2FilterFaultLedIndicator**
- QLedIndicator * **guider2FilterWarningLedIndicator**
- QLedIndicator * **guider2FilterTargetReachedLedIndicator**
- QLedIndicator * **guider2FilterEnabledLedIndicator**
- QLedIndicator * **guider2FilterLimitLedIndicator**
- QLedIndicator * **guider2FilterFollowingErrorLedIndicator**
- QLedIndicator * **guider2FilterSetPointLedIndicator**
- QLabel * **label_120**
- QLabel * **guider2FilterLabel**
- QGroupBox * **groupBox_12**
- QGridLayout * **gridLayout_2**
- QLedIndicator * **wfs1PowerLedIndicator**
- QLabel * **label_8**

- QLabel * **wfs1ProbePositionLabel**
- QLabel * **label_7**
- QLabel * **wfs1ArmPositionLabel**
- QLedIndicator * **wfs1ConfiguredLedIndicator**
- QGroupBox * **groupBox_6**
- QLedIndicator * **wfs1ArmFaultLedIndicator**
- QLedIndicator * **wfs1ArmEnabledLedIndicator**
- QLedIndicator * **wfs1ArmSetPointLedIndicator**
- QLabel * **wfs1ArmEncoderYLabel**
- QLabel * **label_80**
- QLedIndicator * **wfs1ArmTargetReachedLedIndicator**
- QLedIndicator * **wfs1ArmFollowingErrorLedIndicator**
- QLabel * **wfs1ArmAngleLabel**
- QLabel * **label_81**
- QLedIndicator * **wfs1ArmWarningLedIndicator**
- QLedIndicator * **wfs1ArmLimitLedIndicator**
- QGroupBox * **groupBox_5**
- QLedIndicator * **wfs1ProbeFaultLedIndicator**
- QLedIndicator * **wfs1ProbeWarningLedIndicator**
- QLedIndicator * **wfs1ProbeTargetReachedLedIndicator**
- QLedIndicator * **wfs1ProbeEnabledLedIndicator**
- QLedIndicator * **wfs1ProbeLimitLedIndicator**
- QLedIndicator * **wfs1ProbeFollowingErrorLedIndicator**
- QLedIndicator * **wfs1ProbeSetPointLedIndicator**
- QLabel * **label_71**
- QLabel * **wfs1ProbeEncoderXLabel**
- QLabel * **label_72**
- QLabel * **wfs1ProbeAngleLabel**
- QGroupBox * **groupBox_23**
- QGridLayout * **gridLayout_7**
- QLedIndicator * **wfs2PowerLedIndicator**
- QLabel * **label_5**
- QLabel * **wfs2ProbePositionLabel**
- QLabel * **label_6**
- QLabel * **wfs2ArmPositionLabel**
- QLedIndicator * **wfs2ConfiguredLedIndicator**
- QGroupBox * **groupBox_8**
- QGridLayout * **gridLayout_8**
- QLedIndicator * **wfs2ArmFaultLedIndicator**
- QLedIndicator * **wfs2ArmEnabledLedIndicator**
- QLedIndicator * **wfs2ArmLimitLedIndicator**
- QLedIndicator * **wfs2ArmWarningLedIndicator**
- QLabel * **label_98**
- QLabel * **wfs2ArmEncoderYLabel**
- QLedIndicator * **wfs2ArmSetPointLedIndicator**
- QLabel * **wfs2ArmAngleLabel**
- QLabel * **label_99**
- QLedIndicator * **wfs2ArmTargetReachedLedIndicator**
- QLedIndicator * **wfs2ArmFollowingErrorLedIndicator**
- QGroupBox * **groupBox_7**

- QGridLayout * **gridLayout_4**
- QLedIndicator * **wfs2ProbeFaultLedIndicator**
- QLedIndicator * **wfs2ProbeEnabledLedIndicator**
- QLedIndicator * **wfs2ProbeSetPointLedIndicator**
- QLedIndicator * **wfs2ProbeFollowingErrorLedIndicator**
- QLedIndicator * **wfs2ProbeTargetReachedLedIndicator**
- QLabel * **wfs2ProbeAngleLabel**
- QLabel * **label_89**
- QLabel * **wfs2ProbeEncoderXLabel**
- QLedIndicator * **wfs2ProbeWarningLedIndicator**
- QLabel * **label_90**
- QLedIndicator * **wfs2ProbeLimitLedIndicator**

The documentation for this class was generated from the following file:

- gui/build/ui_probeStatusTab.h

9.1017 Ui_productionTab Class Reference

Public Member Functions

- void **setupUi** (QWidget *productionTab)
- void **retranslateUi** (QWidget *productionTab)

Public Attributes

- QTabWidget * **productionTabWidget**
- QWidget * **stackingWidget**
- QTcsFrame * **tcsFrame**
- QLabel * **label_16**
- QWidget * **widget_11**
- QPushButton * **upperCenterPresetsPushButton**
- QPushButton * **leftPresetsPushButton**
- QPushButton * **lowerLeftPresetsPushButton**
- QPushButton * **lowerRightPresetsPushButton**
- QPushButton * **homePresetsPushButton**
- QPushButton * **centerPresetsPushButton**
- QPushButton * **upperLeftPresetsPushButton**
- QPushButton * **upperRightPresetsPushButton**
- QPushButton * **rightPresetsPushButton**
- QTcsFrame * **videoSwitchTcsFrame**
- QGridLayout * **gridLayout_2**
- QLedIndicator * **pictureInPictureVideoLedIndicator**
- QLabel * **label_17**
- QLedIndicator * **ccamVideoLedIndicator**
- QLedIndicator * **waveScopeVideoLedIndicator**
- QLedIndicator * **loadingBayVideoLedIndicator**
- QLedIndicator * **aoaVideoLedIndicator**
- QTcsFrame * **tcsFrame_3**
- QGridLayout * **gridLayout_4**
- QLabel * **label_31**
- QLedIndicator * **waveScopePowerLedIndicator**
- QTcsFrame * **dmiTcsFrame_2**
- QGridLayout * **gridLayout_11**
- QGroupBox * **groupBox_25**
- QGridLayout * **gridLayout_14**
- QLabel * **label_57**
- QLabel * **label_58**
- QLabel * **label_60**
- QLineEdit * **ccasDistanceLineEdit**
- QLineEdit * **ccasPeakAmplLineEdit**
- QLineEdit * **ccasCoupl1LineEdit**
- QCheckBox * **enableCCASMeasurementsCheckBox**
- QLabel * **label_61**
- QLineEdit * **ccasLastValidDistanceLineEdit**
- QLabel * **label_254**
- QLabel * **label_253**

- QLedIndicator * **ccasDMIPipelineRunningLedIndicator**
- QWidget * **operationsWidget**
- QTcsFrame * **pfipControlTcsFrame**
- QGridLayout * **gridLayout_5**
- QPushButton * **selectPV_CWFSCameraPushButton**
- QLabel * **label**
- QLabel * **Guider1FilterPositionLabel**
- QLedIndicator * **HR_SASCameraPowerLedIndicator**
- QPushButton * **yellow_CloseShutterPushButton**
- QLabel * **label_52**
- QLabel * **ShutterPositionLabel**
- QLabel * **label_56**
- QPushButton * **setACQCameraFilterPushButton**
- QFrame * **line**
- QLabel * **label_43**
- QLedIndicator * **ACQCameraPowerLedIndicator**
- QLedIndicator * **BIBCameraPowerLedIndicator**
- QLedIndicator * **WFS1CameraPowerLedIndicator**
- QLabel * **label_34**
- QLedIndicator * **CWFSCameraPowerLedIndicator**
- QLedIndicator * **Guider1CameraPowerLedIndicator**
- QComboBox * **ACQCameraFilterSelectionComboBox**
- QPushButton * **green_OpenShutterPushButton**
- QLabel * **label_51**
- QLabel * **label_37**
- QPushButton * **deployRetractPV_CWFSCameraMirrorPushButton**
- QLedIndicator * **PVCameraPowerLedIndicator**
- QLabel * **label_55**
- QLabel * **label_33**
- QComboBox * **Guider1FilterSelectionComboBox**
- QLabel * **PV_CWFSCameraMirrorPositionLabel**
- QLabel * **PV_CWFSCameraPositionLabel**
- QLedIndicator * **HPF_ACQCameraPowerLedIndicator**
- QPushButton * **setGuider2FilterPushButton**
- QLabel * **label_59**
- QLabel * **Guider2FilterPositionLabel**
- QComboBox * **Guider2FilterSelectionComboBox**
- QPushButton * **deployRetractACQCameraMirrorPushButton**
- QLabel * **ACQCameraFilterPositionLabel**
- QLedIndicator * **Guider2CameraPowerLedIndicator**
- QLabel * **ACQCameraMirrorPositionLabel**
- QPushButton * **setGuider1FilterPushButton**
- QLedIndicator * **WFS2CameraPowerLedIndicator**
- QLabel * **label_54**
- QLabel * **label_1**
- QLedIndicator * **TTCameraPowerLedIndicator**
- QTcsFrame * **tcsFrame_7**
- QGridLayout * **gridLayout_16**
- QPushButton * **red_abortTrackerPushButton**
- QLabel * **label_224**

- QGroupBox * **groupBox_7**
- QGridLayout * **gridLayout_15**
- QLedIndicator * **x_motionLedIndicator**
- QLedIndicator * **y_motionLedIndicator**
- QLedIndicator * **rho_motionLedIndicator**
- QLedIndicator * **hex1_motionLedIndicator**
- QLedIndicator * **hex2_motionLedIndicator**
- QLedIndicator * **hex3_motionLedIndicator**
- QLedIndicator * **hex4_motionLedIndicator**
- QLedIndicator * **hex5_motionLedIndicator**
- QLedIndicator * **hex6_motionLedIndicator**
- QTcsFrame * **tcsFrame_5**
- QGridLayout * **gridLayout_6**
- QLabel * **currentNumPointsLabel**
- QLabel * **nextDecLabel**
- QLabel * **currentAzimuthLabel**
- QLabel * **label_12**
- QLabel * **label_19**
- QLabel * **nextRaLabel**
- QLabel * **nextDurationLabel**
- QLabel * **currentDecLabel**
- QLabel * **label_11**
- QLabel * **label_10**
- QLabel * **nextAzimuthLabel**
- QLabel * **label_14**
- QLabel * **label_7**
- QLabel * **currentTrackStartTimeLabel**
- QLabel * **currentDurationLabel**
- QLabel * **currentRaLabel**
- QLabel * **nextTrackStartTimeLabel**
- QLabel * **label_13**
- QLabel * **currentAltitudeLabel**
- QLabel * **label_9**
- QLabel * **label_8**
- QLabel * **nextNumPointsLabel**
- QLabel * **label_6**
- QTextEdit * **trackProgressTextEdit**
- QLedIndicator * **trajectoryLedIndicator**
- QLabel * **label_20**
- QLabel * **label_15**
- QLabel * **nextAltitudeLabel**
- QLabel * **trackTimeRemainingLabel**
- QLabel * **trackTimeUntilStartLabel**
- QLabel * **trackStartTimeLabel**
- QLabel * **trackStopTimeLabel**
- QFrame * **line_2**
- QLabel * **label_21**
- QTcsFrame * **dmiTcsFrame**
- QLabel * **label_245**
- QGridLayout * **gridLayout_7**

- QLabel * **label_246**
- QLedIndicator * **sacDMIPipelineRunningLedIndicator**
- QGroupBox * **groupBox_26**
- QGridLayout * **gridLayout_20**
- QLabel * **label_247**
- QLabel * **label_248**
- QLineEdit * **sacDistanceLineEdit**
- QLineEdit * **sacPeakAmplLineEdit**
- QLineEdit * **sacCoupl1LineEdit**
- QLabel * **label_249**
- QLabel * **label_250**
- QLineEdit * **sacLastValidDistanceLineEdit**
- QCheckBox * **enableSACMeasurementsCheckBox**
- QTcsFrame * **tcsFrame_4**
- QTabWidget * **offsetTypeTabWidget**
- QWidget * **tab_4**
- QLabel * **label_177**
- QWidget * **widget_28**
- QLineEdit * **tipTilt_thetaPhiDeltaLineEdit**
- QLabel * **label_165**
- QLabel * **label_166**
- QLabel * **label_167**
- QLabel * **label_168**
- QLabel * **label_169**
- QPushButton * **tipTilt_plusThetaPushButton**
- QPushButton * **tipTilt_minusThetaPushButton**
- QPushButton * **tipTilt_minusPhiPushButton**
- QPushButton * **tipTilt_plusPhiPushButton**
- QWidget * **tab_3**
- QLabel * **label_2**
- QWidget * **widget_7**
- QLineEdit * **RaDecDeltaLineEdit**
- QLabel * **label_32**
- QLabel * **label_35**
- QLabel * **label_36**
- QLabel * **label_38**
- QLabel * **label_44**
- QPushButton * **plusDecPushButton**
- QPushButton * **minusDecPushButton**
- QPushButton * **minusRaPushButton**
- QPushButton * **plusRaPushButton**
- QLabel * **RaOffsetLabel**
- QLabel * **label_106**
- QLabel * **label_105**
- QLabel * **DecOffsetLabel**
- QPushButton * **resetRaDecCumulativeOffsetPushButton**
- QCheckBox * **raDecCorrectionCheckBox**
- QWidget * **tab_5**
- QWidget * **widget_9**
- QLineEdit * **xyDeltaLineEdit**

- QLabel * **label_39**
- QLabel * **label_40**
- QLabel * **label_41**
- QLabel * **label_42**
- QLabel * **label_45**
- QPushButton * **plusYPushButton**
- QPushButton * **minusYPushButton**
- QPushButton * **minusXPushButton**
- QPushButton * **plusXPushButton**
- QLabel * **label_3**
- QCheckBox * **xyCorrectionCheckBox**
- QWidget * **tab_6**
- QWidget * **widget_10**
- QLineEdit * **rhoDeltaLineEdit**
- QLabel * **label_46**
- QLabel * **label_47**
- QLabel * **label_48**
- QPushButton * **plusRhoPushButton**
- QPushButton * **minusRhoPushButton**
- QLabel * **label_4**
- QWidget * **widget_8**
- QLineEdit * **focusDeltaLineEdit**
- QLabel * **label_49**
- QLabel * **label_50**
- QLabel * **label_53**
- QPushButton * **plusFocusPushButton**
- QPushButton * **minusFocusPushButton**
- QLabel * **label_5**
- QLabel * **label_18**
- QTcsFrame * **tcsFrame_2**
- QGridLayout * **gridLayout_8**
- QLabel * **label_22**
- QSpacerItem * **horizontalSpacer**
- QPushButton * **green_loadPushButton**
- QTabWidget * **trajectoryTabWidget**
- QWidget * **tab_9**
- QLabel * **label_70**
- QLineEdit * **raDecRaLineEdit**
- QLineEdit * **raDecDecLineEdit**
- QLabel * **label_69**
- QLineEdit * **raDecAzLineEdit**
- QLabel * **label_87**
- QGroupBox * **groupBox**
- QGridLayout * **gridLayout**
- QRadioButton * **raDecEastRadioButton**
- QRadioButton * **raDecWestRadioButton**
- QLineEdit * **raDecEquinoxLineEdit**
- QLabel * **label_91**
- QLineEdit * **raDecEpochLineEdit**
- QLabel * **label_92**

- QGroupBox * **groupBox_4**
- QPushButton * **raDecEastTrajectoryPushButton**
- QPushButton * **raDecSouthTrajectoryPushButton**
- QPushButton * **raDecWestTrajectoryPushButton**
- QWidget * **tab_10**
- QLabel * **label_71**
- QLineEdit * **xyYLineEdit**
- QLineEdit * **xyXLineEdit**
- QLabel * **label_72**
- QLineEdit * **xyAzLineEdit**
- QLabel * **label_94**
- QWidget * **tab_15**
- QLineEdit * **fileFileLineEdit**
- QLabel * **label_74**
- QLineEdit * **fileRaLineEdit**
- QLineEdit * **fileDecLineEdit**
- QLabel * **label_73**
- QLabel * **label_76**
- QPushButton * **fileFileBrowsePushButton**
- QWidget * **tab**
- QLabel * **label_75**
- QLineEdit * **satFileLineEdit**
- QLabel * **label_88**
- QLineEdit * **satObjectLineEdit**
- QLabel * **label_78**
- QLineEdit * **satAzLineEdit**
- QPushButton * **satFileBrowsePushButton**
- QWidget * **tab_2**
- QLabel * **label_89**
- QLabel * **label_77**
- QLineEdit * **ephemFileLineEdit**
- QGroupBox * **groupBox_2**
- QGridLayout * **gridLayout_3**
- QRadioButton * **ephemEastRadioButton**
- QRadioButton * **ephemWestRadioButton**
- QLineEdit * **ephemAzLineEdit**
- QPushButton * **ephemFileBrowsePushButton**
- QGroupBox * **groupBox_1**
- QGridLayout * **gridLayout_9**
- QGroupBox * **groupBox_3**
- QGridLayout * **gridLayout_10**
- QCheckBox * **moveDomeCheckBox**
- QCheckBox * **moveStructureCheckBox**
- QCheckBox * **moveProbesCheckBox**
- QCheckBox * **correctParallacticCheckBox**
- QPushButton * **yellow_goNextPushButton**
- QPushButton * **correctForStructurePushButton**
- QPushButton * **correctForParallacticPushButton**
- QSpacerItem * **verticalSpacer_2**
- QPushButton * **red_cancelPushButton**

- QTcsFrame * **tcsFrame_6**
- QGridLayout * **gridLayout_12**
- QLabel * **label_225**
- QCheckBox * **ACQStoringCheckBox**
- QCheckBox * **HPF_ACQStoringCheckBox**
- QCheckBox * **Guider1StoringCheckBox**
- QCheckBox * **Guider2StoringCheckBox**
- QPushButton * **green_enableAllStoringPushButton**
- QCheckBox * **WFS1StoringCheckBox**
- QCheckBox * **WFS2StoringCheckBox**
- QCheckBox * **CWFSStoringCheckBox**
- QCheckBox * **PVStoringCheckBox**
- QCheckBox * **BIBStoringCheckBox**
- QCheckBox * **TTStoringCheckBox**
- QPushButton * **yellow_disableAllStoringPushButton**
- QTcsFrame * **tcsFrame_8**
- QGridLayout * **gridLayout_13**
- QCheckBox * **Guider2StartPipelineAndImagesCheckBox**
- QCheckBox * **WFS1StartPipelineAndImagesCheckBox**
- QCheckBox * **PVStartPipelineAndImagesCheckBox**
- QCheckBox * **CWFSStartPipelineAndImagesCheckBox**
- QCheckBox * **WFS2StartPipelineAndImagesCheckBox**
- QCheckBox * **BIBStartPipelineAndImagesCheckBox**
- QCheckBox * **TTStartPipelineAndImagesCheckBox**
- QLabel * **label_255**
- QCheckBox * **ACQStartPipelineAndImagesCheckBox**
- QCheckBox * **HPF_ACQStartPipelineAndImagesCheckBox**
- QCheckBox * **Guider1StartPipelineAndImagesCheckBox**
- QPushButton * **green_startPipelineAndImagesPushButton**
- QPushButton * **yellow_stopPipelineAndImagesPushButton**
- QTcsFrame * **domeTcsFrame**
- QGridLayout * **gridLayout_17**
- QLedIndicator * **domeMovingLedIndicator**
- QLabel * **label_238**
- QLedIndicator * **domePMACCommsLedIndicator**
- QLabel * **label_234**
- QPushButton * **green_moveDomeAbsPushButton**
- QLabel * **label_227**
- QBlinkingLineEdit * **domePositionBlinkingLineEdit**
- QLabel * **label_228**
- QLedIndicator * **domeStatusValidLedIndicator**
- QLabel * **label_226**
- QLineEdit * **absDomeTargetLineEdit**
- QLabel * **label_229**
- QLabel * **label_233**
- QLabel * **label_252**
- QLineEdit * **domeTargetLineEdit**
- QPushButton * **ccasDomePushButton**
- QPushButton * **yellow_stopDomePushButton**
- QPushButton * **red_abortDomePushButton**

- QTcsFrame * **shutterTcsFrame**
- QGridLayout * **gridLayout_18**
- QLabel * **label_251**
- QLineEdit * **shutterTargetLineEdit**
- QLabel * **label_236**
- QPushButton * **green_openShutterPushButton**
- QPushButton * **yellow_closeShutterPushButton**
- QPushButton * **yellow_stopShutterPushButton**
- QLedIndicator * **shutterMovingLedIndicator**
- QLabel * **label_230**
- QLedIndicator * **shutterStatusValidLedIndicator**
- QLabel * **label_235**
- QLedIndicator * **shutterPMACCommsLedIndicator**
- QLabel * **label_232**
- QPushButton * **red_abortShutterPushButton**
- QLabel * **label_231**
- QBlinkingLineEdit * **shutterPositionBlinkingLineEdit**
- QLabel * **label_62**
- QBlinkingLineEdit * **shutterMotionBlinkingLineEdit**
- QTcsFrame * **structureTcsFrame**
- QGridLayout * **gridLayout_19**
- QLedIndicator * **structureInControlLedIndicator**
- QLedIndicator * **structureStatusValidLedIndicator**
- QLabel * **label_241**
- QLabel * **label_239**
- QLabel * **label_242**
- QLedIndicator * **structureMovingLedIndicator**
- QLabel * **label_243**
- QPushButton * **green_moveStructureToTrueAzPushButton**
- QLineEdit * **moveStructureTargetLineEdit**
- QLabel * **label_237**
- QBlinkingLineEdit * **structureAzimuthBlinkingLineEdit**
- QLabel * **label_244**
- QLineEdit * **structureTargetLineEdit**
- QLabel * **label_240**
- QPushButton * **yellow_stopStructurePushButton**
- QPushButton * **ccasStructurePushButton**
- QButtonGroup * **raDecDirectionButtonGroup**
- QButtonGroup * **raDecButtonGroup**
- QButtonGroup * **ephemDirectionButtonGroup**
- QButtonGroup * **rhoButtonGroup**
- QButtonGroup * **focusButtonGroup**
- QButtonGroup * **tipTiltButtonGroup**
- QButtonGroup * **presetsButtonGroup**
- QButtonGroup * **xyButtonGroup**

The documentation for this class was generated from the following file:

- gui/build/ui_productionTab.h

9.1018 Ui_shutterTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗shutterTab)
- void **retranslateUi** (QWidget ∗shutterTab)

Public Attributes

- QGridLayout ∗ **gridLayout**
- QTcsFrame ∗ **shutterTcsFrame**
- QGridLayout ∗ **gridLayout_2**
- QLabel ∗ **label**
- QGroupBox ∗ **groupBox_15**
- QGridLayout ∗ **gridLayout_8**
- QPushButton ∗ **yellow_stopShutterPushButton**
- QPushButton ∗ **green_openShutterPushButton**
- QPushButton ∗ **yellow_closeShutterPushButton**
- QPushButton ∗ **red_abortShutterPushButton**
- QPushButton ∗ **red_resetShutterPushButton**
- QGroupBox ∗ **groupBox_16**
- QGridLayout ∗ **gridLayout_3**
- QLabel ∗ **label_31**
- QBlinkingLineEdit ∗ **shutterPositionBlinkingLineEdit**
- QLabel ∗ **label_34**
- QBlinkingLineEdit ∗ **shutterMotionBlinkingLineEdit**
- QLabel ∗ **label_24**
- QLineEdit ∗ **shutterTargetLineEdit**
- QLabel ∗ **label_29**
- QLedIndicator ∗ **shutterMovingLedIndicator**
- QLabel ∗ **label_30**
- QLedIndicator ∗ **shutterPMACCommsLedIndicator**
- QLabel ∗ **label_26**
- QLineEdit ∗ **shutterErrorNumLineEdit**
- QLabel ∗ **label_32**
- QLineEdit ∗ **shutterLocalRemoteLineEdit**
- QLabel ∗ **label_44**
- QLedIndicator ∗ **shutterStatusValidLedIndicator**
- QSpacerItem ∗ **verticalSpacer**

The documentation for this class was generated from the following file:

- gui/build/ui_shutterTab.h

9.1019 Ui_structureTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗structureTab)
- void **retranslateUi** (QWidget ∗structureTab)

Public Attributes

- QGridLayout ∗ **gridLayout_2**
- QTcsFrame ∗ **structureTcsFrame**
- QLabel ∗ **label**
- QGroupBox ∗ **groupBox_18**
- QGridLayout ∗ **gridLayout_6**
- QLedIndicator ∗ **structureStatusValidLedIndicator**
- QLabel ∗ **label_33**
- QBlinkingLineEdit ∗ **structureAzimuthBlinkingLineEdit**
- QLabel ∗ **label_27**
- QLabel ∗ **label_25**
- QLedIndicator ∗ **structureMovingLedIndicator**
- QLabel ∗ **label_36**
- QLabel ∗ **label_35**
- QLedIndicator ∗ **structureInControlLedIndicator**
- QLineEdit ∗ **structureTargetLineEdit**
- QGroupBox ∗ **groupBox_19**
- QGridLayout ∗ **gridLayout_4**
- QLabel ∗ **label_38**
- QLineEdit ∗ **structureFootHeightsLineEdit**
- QGroupBox ∗ **groupBox_17**
- QGridLayout ∗ **gridLayout_9**
- QLabel ∗ **label_20**
- QPushButton ∗ **ccasStructurePushButton**
- QPushButton ∗ **stowStructurePushButton**
- QPushButton ∗ **yellow_stopStructurePushButton**
- QPushButton ∗ **moveStructureToStructAzPushButton**
- QPushButton ∗ **green_moveStructureToTrueAzPushButton**
- QLineEdit ∗ **moveStructureTargetLineEdit**

The documentation for this class was generated from the following file:

- gui/build/ui_structureTab.h

9.1020 Ui_trackerStatusTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗trackerStatusTab)
- void **retranslateUi** (QWidget ∗trackerStatusTab)

Public Attributes

- QLedIndicator ∗ **servo_cabinet_line_monitorLedIndicator**
- QLedIndicator ∗ **trajectoryLedIndicator**
- QLabel ∗ **label_3**
- QLabel ∗ **idx_timeLabel**
- QTabWidget ∗ **trackerStatusTabWidget**
- QWidget ∗ **tab**
- QFrame ∗ **frame**
- QGridLayout ∗ **gridLayout_3**
- QLabel ∗ **label_13**
- QLabel ∗ **label_8**
- QLabel ∗ **x_rtf_mmLabel**
- QLabel ∗ **label_16**
- QLabel ∗ **label_7**
- QLabel ∗ **label_23**
- QLabel ∗ **rho_rtf_degLabel**
- QLabel ∗ **y_rtf_mmLabel**
- QLabel ∗ **phi_rtf_degLabel**
- QLabel ∗ **label_15**
- QLabel ∗ **label_11**
- QLabel ∗ **theta_rtf_degLabel**
- QLabel ∗ **z_rtf_mmLabel**
- QFrame ∗ **frame_8**
- QGridLayout ∗ **gridLayout_2**
- QLabel ∗ **label_44**
- QLedIndicator ∗ **y_track_warningLedIndicator**
- QLabel ∗ **label_52**
- QLedIndicator ∗ **ux_track_torque_limit_swerLedIndicator**
- QSpacerItem ∗ **verticalSpacer_40**
- QLedIndicator ∗ **y_slew_warningLedIndicator**
- QLedIndicator ∗ **hex5_warningLedIndicator**
- QLedIndicator ∗ **y_slew_errorLedIndicator**
- QLedIndicator ∗ **ux_track_warningLedIndicator**
- QLabel ∗ **label_42**
- QLedIndicator ∗ **hex2_errorLedIndicator**
- QLedIndicator ∗ **hex2_warningLedIndicator**
- QLabel ∗ **label_257**
- QLedIndicator ∗ **lx_slew_warningLedIndicator**
- QLedIndicator ∗ **hex6_warningLedIndicator**
- QLedIndicator ∗ **ux_slew_torque_limit_swerLedIndicator**
- QLabel ∗ **label_45**
- QLedIndicator ∗ **hex4_warningLedIndicator**

- QLedIndicator * **lx_slew_torque_limit_swerLedIndicator**
- QLabel * **label_51**
- QLedIndicator * **lx_track_warningLedIndicator**
- QLedIndicator * **lx_track_errorLedIndicator**
- QLedIndicator * **hex6_errorLedIndicator**
- QLedIndicator * **hex1_warningLedIndicator**
- QLabel * **label_5**
- QLedIndicator * **y_track_torque_limit_swerLedIndicator**
- QLedIndicator * **hex5_errorLedIndicator**
- QSpacerItem * **verticalSpacer_16**
- QLedIndicator * **y_slew_torque_limit_swerLedIndicator**
- QLedIndicator * **hex3_errorLedIndicator**
- QLedIndicator * **rho_warningLedIndicator**
- QLedIndicator * **hex4_errorLedIndicator**
- QLedIndicator * **hex1_errorLedIndicator**
- QLedIndicator * **y_track_errorLedIndicator**
- QLedIndicator * **lx_slew_errorLedIndicator**
- QLabel * **label_9**
- QLabel * **label_43**
- QLabel * **label_53**
- QSpacerItem * **verticalSpacer_14**
- QLedIndicator * **cfd_warningLedIndicator**
- QLedIndicator * **rho_errorLedIndicator**
- QLedIndicator * **ux_slew_errorLedIndicator**
- QSpacerItem * **verticalSpacer_39**
- QLedIndicator * **hex3_warningLedIndicator**
- QLedIndicator * **cfd_errorLedIndicator**
- QLedIndicator * **ux_track_errorLedIndicator**
- QLedIndicator * **lx_track_torque_limit_swerLedIndicator**
- QSpacerItem * **verticalSpacer_15**
- QLedIndicator * **ux_slew_warningLedIndicator**
- QSpacerItem * **verticalSpacer_41**
- QSpacerItem * **verticalSpacer_42**
- QFrame * **frame_3**
- QGridLayout * **gridLayout_9**
- QLabel * **hex5_tsf_cmd_mmLabel**
- QLabel * **label_86**
- QLabel * **hex6_tsf_cmd_mmLabel**
- QLabel * **label_112**
- QLabel * **y_tsf_cmd_mmLabel**
- QLabel * **label_111**
- QLabel * **label_93**
- QLabel * **hex4_tsf_cmd_mmLabel**
- QLabel * **label_83**
- QLabel * **label_84**
- QLabel * **hex3_tsf_cmd_mmLabel**
- QLabel * **hex1_tsf_cmd_mmLabel**
- QLabel * **hex2_tsf_cmd_mmLabel**
- QLabel * **ux_tsf_cmd_mmLabel**
- QLabel * **lx_tsf_cmd_mmLabel**

- QLabel * **label_87**
- QLabel * **label_91**
- QLabel * **label_89**
- QLabel * **label_85**
- QLabel * **label_32**
- QLabel * **rho_tsf_cmd_degLabel**
- QFrame * **frame_14**
- QGridLayout * **gridLayout_7**
- QLabel * **label_113**
- QLabel * **label_50**
- QLabel * **label_81**
- QLabel * **hex5_fe_mmLabel**
- QLabel * **label_88**
- QLabel * **ux_fe_mmLabel**
- QLabel * **label_90**
- QLabel * **label_49**
- QLabel * **label_92**
- QLabel * **label_46**
- QLabel * **label_63**
- QLabel * **hex1_fe_mmLabel**
- QLabel * **hex6_fe_mmLabel**
- QLabel * **y_fe_mmLabel**
- QLabel * **rho_fe_degLabel**
- QLabel * **label_47**
- QLabel * **lx_fe_mmLabel**
- QLabel * **label_48**
- QLabel * **hex3_fe_mmLabel**
- QLabel * **hex4_fe_mmLabel**
- QLabel * **hex2_fe_mmLabel**
- QFrame * **frame_16**
- QGridLayout * **gridLayout_10**
- QLabel * **label_123**
- QLabel * **cfd_load_cellLabel**
- QLabel * **label_125**
- QLabel * **y_track_brake_load_cellLabel**
- QFrame * **frame_2**
- QGridLayout * **gridLayout_4**
- QLedIndicator * **rho_lock_outLedIndicator**
- QLedIndicator * **hex3_enaLedIndicator**
- QLedIndicator * **hex1_enaLedIndicator**
- QLedIndicator * **hex3_rtoLedIndicator**
- QLedIndicator * **y_slew_pwrLedIndicator**
- QLedIndicator * **rho_pwrLedIndicator**
- QLedIndicator * **lx_slew_pwrLedIndicator**
- QLedIndicator * **lx_track_pwrLedIndicator**
- QLedIndicator * **hex3_servoLedIndicator**
- QLedIndicator * **lx_track_modeLedIndicator**
- QLedIndicator * **ux_track_servoLedIndicator**
- QLedIndicator * **lx_track_brake_pressureLedIndicator**
- QLabel * **label_122**

- QLedIndicator * **ux_slew_modeLedIndicator**
- QSpacerItem * **verticalSpacer_12**
- QLedIndicator * **ux_track_enaLedIndicator**
- QLabel * **label_20**
- QSpacerItem * **verticalSpacer_13**
- QLabel * **label_24**
- QLedIndicator * **y_track_servoLedIndicator**
- QLedIndicator * **y_track_modeLedIndicator**
- QLedIndicator * **hex4_rtoLedIndicator**
- QLedIndicator * **hex2_enaLedIndicator**
- QLedIndicator * **y_track_brake_pressureLedIndicator**
- QSpacerItem * **verticalSpacer_7**
- QLedIndicator * **y_slew_modeLedIndicator**
- QLedIndicator * **ux_track_modeLedIndicator**
- QLedIndicator * **y_track_enaLedIndicator**
- QLedIndicator * **lx_slew_modeLedIndicator**
- QLabel * **label_25**
- QLedIndicator * **rho_rtoLedIndicator**
- QLedIndicator * **hex1_track_pwrLedIndicator**
- QLedIndicator * **hex6_lock_outLedIndicator**
- QLedIndicator * **hex4_enaLedIndicator**
- QLedIndicator * **x_track_lock_outLedIndicator**
- QLedIndicator * **hex6_pwrLedIndicator**
- QLabel * **label_6**
- QLedIndicator * **hex6_servoLedIndicator**
- QLabel * **label_29**
- QLedIndicator * **hex3_lock_outLedIndicator**
- QLedIndicator * **hex5_track_rtoLedIndicator**
- QLedIndicator * **hex2_lock_outLedIndicator**
- QLedIndicator * **hex6_rtoLedIndicator**
- QLedIndicator * **ux_slew_enaLedIndicator**
- QLedIndicator * **rho_enaLedIndicator**
- QLedIndicator * **hex5_lock_outLedIndicator**
- QLedIndicator * **cfd_servoLedIndicator**
- QLedIndicator * **hex5_pwrLedIndicator**
- QLedIndicator * **y_track_rtoLedIndicator**
- QLedIndicator * **lx_slew_rtoLedIndicator**
- QLabel * **label_218**
- QLabel * **label_18**
- QLedIndicator * **ux_slew_servoLedIndicator**
- QLedIndicator * **y_track_pwrLedIndicator**
- QLedIndicator * **hex1_lock_outLedIndicator**
- QLedIndicator * **y_slew_enaLedIndicator**
- QLabel * **label_22**
- QLedIndicator * **y_slew_rtoLedIndicator_**
- QLedIndicator * **lx_track_enaLedIndicator**
- QLabel * **label_28**
- QLedIndicator * **hex1_track_rtoLedIndicator**
- QLedIndicator * **hex2_servoLedIndicator**
- QLabel * **label_4**

- QLedIndicator * **rho_servoLedIndicator**
- QSpacerItem * **verticalSpacer_8**
- QLedIndicator * **y_slew_brake_pressureLedIndicator**
- QLedIndicator * **ux_track_pwrLedIndicator**
- QLedIndicator * **ux_slew_brake_pressureLedIndicator**
- QLedIndicator * **hex3_pwrLedIndicator**
- QLedIndicator * **hex5_enaLedIndicator**
- QLedIndicator * **cfd_pwrLedIndicator**
- QSpacerItem * **verticalSpacer_10**
- QLedIndicator * **lx_slew_enaLedIndicator**
- QLedIndicator * **hex2_track_rtoLedIndicator**
- QLedIndicator * **y_slew_servoLedIndicator**
- QLedIndicator * **ux_slew_pwrLedIndicator**
- QLedIndicator * **cfd_rtoLedIndicator**
- QLedIndicator * **hex5_servoLedIndicator**
- QLedIndicator * **lx_slew_servoLedIndicator**
- QSpacerItem * **verticalSpacer_6**
- QLedIndicator * **x_slew_lock_outLedIndicator**
- QLedIndicator * **lx_slew_brake_pressureLedIndicator**
- QLabel * **label_27**
- QLabel * **label_26**
- QLedIndicator * **hex4_lock_outLedIndicator**
- QSpacerItem * **verticalSpacer_11**
- QLedIndicator * **hex4_pwrLedIndicator**
- QLedIndicator * **hex6_enaLedIndicator**
- QLedIndicator * **hex2_pwrLedIndicator**
- QLedIndicator * **hex4_servoLedIndicator**
- QLedIndicator * **ux_track_rtoLedIndicator**
- QLedIndicator * **y_lock_outLedIndicator**
- QLedIndicator * **lx_track_servoLedIndicator**
- QLedIndicator * **hex1_servoLedIndicator**
- QLedIndicator * **ux_track_brake_pressureLedIndicator**
- QLedIndicator * **ux_slew_rtoLedIndicator**
- QLabel * **label_21**
- QLedIndicator * **lx_track_rtoLedIndicator**
- QFrame * **frame_13**
- QGridLayout * **gridLayout_5**
- QLabel * **hex6_tsf_mmLabel**
- QLabel * **hex4_tsf_mmLabel**
- QLabel * **y_tsf_mmLabel**
- QLabel * **label_33**
- QLabel * **hex5_tsf_mmLabel**
- QLabel * **label_10**
- QLabel * **rho_tsf_degLabel**
- QLabel * **hex2_tsf_mmLabel**
- QLabel * **hex3_tsf_mmLabel**
- QLabel * **label_12**
- QLabel * **label_17**
- QLabel * **label_19**
- QLabel * **label_66**

- QLabel * **label_74**
- QLabel * **hex1_tsf_mmLabel**
- QLabel * **label_69**
- QLabel * **lx_tsf_mmLabel**
- QLabel * **label_62**
- QLabel * **ux_tsf_mmLabel**
- QLabel * **label_72**
- QLabel * **label_70**
- QFrame * **frame_17**
- QGridLayout * **gridLayout_11**
- QLabel * **label_126**
- QLabel * **cfd_speedLabel**
- QLabel * **label_127**
- QLabel * **y_slew_brake_load_cellLabel**
- QFrame * **frame_7**
- QGridLayout * **gridLayout_32**
- QLabel * **label_36**
- QLabel * **label_37**
- QLabel * **label_35**
- QLedIndicator * **ux_left_soft_limit_swLedIndicator**
- QLedIndicator * **ux_right_soft_limit_swLedIndicator**
- QLabel * **label_38**
- QLedIndicator * **ux_left_hard_limit_swLedIndicator**
- QLedIndicator * **ux_right_hard_limit_swLedIndicator**
- QLabel * **label_41**
- QLedIndicator * **lx_left_soft_limit_swLedIndicator**
- QLedIndicator * **lx_right_soft_limit_swLedIndicator**
- QLabel * **label_40**
- QLedIndicator * **lx_left_hard_limit_swLedIndicator**
- QLedIndicator * **lx_right_hard_limit_swLedIndicator**
- QLabel * **label_34**
- QLedIndicator * **y_bottom_soft_limit_swLedIndicator**
- QLedIndicator * **y_top_soft_limit_swLedIndicator**
- QLabel * **label_31**
- QLedIndicator * **y_bottom_hard_limit_swLedIndicator**
- QLedIndicator * **y_top_hard_limit_swLedIndicator**
- QLabel * **label_39**
- QLedIndicator * **rho_cw_hard_limit_swLedIndicator**
- QLedIndicator * **rho_ccw_hard_limit_swLedIndicator**
- QLedIndicator * **rho_soft_limit_swLedIndicator**
- QLabel * **label**
- QLedIndicator * **hex1_bottom_hard_limit_swLedIndicator**
- QLedIndicator * **hex1_top_hard_limit_swLedIndicator**
- QSpacerItem * **verticalSpacer**
- QLedIndicator * **hex2_bottom_hard_limit_swLedIndicator**
- QLedIndicator * **hex2_top_hard_limit_swLedIndicator**
- QSpacerItem * **verticalSpacer_2**
- QLedIndicator * **hex3_bottom_hard_limit_swLedIndicator**
- QLedIndicator * **hex3_top_hard_limit_swLedIndicator**
- QSpacerItem * **verticalSpacer_3**

- QLedIndicator * **hex4_bottom_hard_limit_swLedIndicator**
- QLedIndicator * **hex4_top_hard_limit_swLedIndicator**
- QSpacerItem * **verticalSpacer_4**
- QLedIndicator * **hex5_bottom_hard_limit_swLedIndicator**
- QLedIndicator * **hex5_top_hard_limit_swLedIndicator**
- QSpacerItem * **verticalSpacer_5**
- QLedIndicator * **hex6_bottom_hard_limit_swLedIndicator**
- QLedIndicator * **hex6_top_hard_limit_swLedIndicator**
- QSpacerItem * **verticalSpacer_9**
- QLabel * **label_2**
- QLedIndicator * **y_screw_proximityLedIndicator**
- QLabel * **label_54**
- QWidget * **tab_2**
- QGroupBox * **groupBox_2**
- QGridLayout * **gridLayout_13**
- QLabel * **label_95**
- QLabel * **ux_track_motor_currentLabel**
- QLabel * **label_96**
- QLabel * **ux_slew_motor_currentLabel**
- QLabel * **label_97**
- QLabel * **lx_track_motor_currentLabel**
- QLabel * **label_98**
- QLabel * **lx_slew_motor_currentLabel**
- QLabel * **label_99**
- QLabel * **y_track_motor_currentLabel**
- QLabel * **label_100**
- QLabel * **y_slew_motor_currentLabel**
- QLabel * **label_107**
- QLabel * **rho_motor_currentLabel**
- QLabel * **label_101**
- QLabel * **hex1_motor_currentLabel**
- QLabel * **label_102**
- QLabel * **hex2_motor_currentLabel**
- QLabel * **label_103**
- QLabel * **hex3_motor_currentLabel**
- QLabel * **label_104**
- QLabel * **hex4_motor_currentLabel**
- QLabel * **label_105**
- QLabel * **hex5_motor_currentLabel**
- QLabel * **label_106**
- QLabel * **hex6_motor_currentLabel**
- QLabel * **label_108**
- QLabel * **cfd_motor_currentLabel**
- QGroupBox * **groupBox_7**
- QGridLayout * **gridLayout_33**
- QLabel * **ux_abs_mmLabel**
- QLabel * **label_82**
- QLabel * **y_abs_mmLabel**
- QLabel * **label_77**
- QLabel * **rho_abs_degLabel**

- QLabel * **label_94**
- QLabel * **label_78**
- QLabel * **lx_abs_mmLabel**
- QSpacerItem * **verticalSpacer_44**
- QSpacerItem * **verticalSpacer_43**
- QSpacerItem * **verticalSpacer_45**
- QLedIndicator * **emergency_stopLedIndicator**
- QGroupBox * **groupBox_11**
- QGridLayout * **gridLayout_12**
- QLedIndicator * **lx_skew_sensor_1LedIndicator**
- QLedIndicator * **lx_skew_sensor_2LedIndicator**
- QLedIndicator * **ux_skew_sensor_1LedIndicator**
- QLedIndicator * **ux_skew_sensor_2LedIndicator**
- QLedIndicator * **movingLedIndicator**
- QLedIndicator * **initingLedIndicator**
- QPushButton * **green_initPushButton**
- QGroupBox * **groupBox_12**
- QGridLayout * **gridLayout_14**
- QLedIndicator * **y_init_stateLedIndicator**
- QLedIndicator * **rho_init_stateLedIndicator**
- QLedIndicator * **ux_init_stateLedIndicator**
- QLedIndicator * **lx_init_stateLedIndicator**
- QLabel * **label_14**
- QLabel * **label_30**
- QLabel * **label_57**
- QLabel * **init_remainingLabel**
- QBlinkingLineEdit * **last_warningBlinkingLineEdit**
- QBlinkingLineEdit * **last_errorBlinkingLineEdit**
- QPushButton * **clear_errorsPushButton**

The documentation for this class was generated from the following file:

- gui/build/ui_trackerStatusTab.h

9.1021 Ui_trackerTab Class Reference

Public Member Functions

- void **setupUi** (QWidget *trackerTab)
- void **retranslateUi** (QWidget *trackerTab)

Public Attributes

- QTcsFrame * **tcsFrame**
- QTabWidget * **moveTypeTabWidget**
- QWidget * **tab**
- QTabWidget * **relativeCoordTypeTabWidget**
- QWidget * **tab_9**
- QLabel * **label_163**
- QLabel * **rel_itf_x_mmLabel**
- QLabel * **label_164**
- QWidget * **widget_28**
- QLineEdit * **itf_thetaPhiDeltaLineEdit**
- QLabel * **label_165**
- QLabel * **label_166**
- QLabel * **label_167**
- QLabel * **label_168**
- QLabel * **label_169**
- QPushButton * **itf_plusThetaPushButton**
- QPushButton * **itf_minusThetaPushButton**
- QPushButton * **itf_minusPhiPushButton**
- QPushButton * **itf_plusPhiPushButton**
- QLabel * **label_170**
- QLabel * **label_171**
- QLabel * **label_172**
- QLabel * **rel_itf_rho_degLabel**
- QWidget * **widget_29**
- QLineEdit * **itf_rhoDeltaLineEdit**
- QLabel * **label_173**
- QLabel * **label_174**
- QLabel * **label_175**
- QPushButton * **itf_plusRhoPushButton**
- QPushButton * **itf_minusRhoPushButton**
- QLabel * **label_176**
- QLabel * **rel_itf_y_mmLabel**
- QLabel * **label_177**
- QLabel * **rel_itf_phi_degLabel**
- QWidget * **widget_30**
- QLineEdit * **itf_zDeltaLineEdit**
- QLabel * **label_178**
- QLabel * **label_179**
- QLabel * **label_180**
- QPushButton * **itf_plusZPushButton**
- QPushButton * **itf_minusZPushButton**

- QLabel * **label_181**
- QLabel * **rel_itf_theta_degLabel**
- QLabel * **rel_itf_z_mmLabel**
- QWidget * **widget_31**
- QLineEdit * **itf_xyDeltaLineEdit**
- QLabel * **label_182**
- QLabel * **label_183**
- QLabel * **label_184**
- QLabel * **label_185**
- QLabel * **label_186**
- QPushButton * **itf_plusYPushButton**
- QPushButton * **itf_minusYPushButton**
- QPushButton * **itf_minusXPushButton**
- QPushButton * **itf_plusXPushButton**
- QWidget * **tab_4**
- QWidget * **widget_3**
- QLineEdit * **rtf_xyDeltaLineEdit**
- QLabel * **label_18**
- QLabel * **label_19**
- QLabel * **label_20**
- QLabel * **label_21**
- QLabel * **label_22**
- QPushButton * **rtf_plusYPushButton**
- QPushButton * **rtf_minusYPushButton**
- QPushButton * **rtf_minusXPushButton**
- QPushButton * **rtf_plusXPushButton**
- QLabel * **label_23**
- QWidget * **widget_4**
- QLineEdit * **rtf_thetaPhiDeltaLineEdit**
- QLabel * **label_24**
- QLabel * **label_25**
- QLabel * **label_26**
- QLabel * **label_27**
- QLabel * **label_28**
- QPushButton * **rtf_plusThetaPushButton**
- QPushButton * **rtf_minusThetaPushButton**
- QPushButton * **rtf_minusPhiPushButton**
- QPushButton * **rtf_plusPhiPushButton**
- QLabel * **label_29**
- QWidget * **widget_5**
- QLineEdit * **rtf_zDeltaLineEdit**
- QLabel * **label_30**
- QLabel * **label_32**
- QLabel * **label_38**
- QPushButton * **rtf_plusZPushButton**
- QPushButton * **rtf_minusZPushButton**
- QLabel * **label_35**
- QWidget * **widget_6**
- QLineEdit * **rtf_rhoDeltaLineEdit**
- QLabel * **label_39**

- QLabel * **label_40**
- QLabel * **label_41**
- QPushButton * **rtf_plusRhoPushButton**
- QPushButton * **rtf_minusRhoPushButton**
- QLabel * **label_42**
- QLabel * **rel_rtf_x_mmLabel**
- QLabel * **rel_rtf_z_mmLabel**
- QLabel * **rel_rtf_rho_degLabel**
- QLabel * **rel_rtf_theta_degLabel**
- QLabel * **label_105**
- QLabel * **rel_rtf_y_mmLabel**
- QLabel * **label_106**
- QLabel * **label_48**
- QLabel * **rel_rtf_phi_degLabel**
- QLabel * **label_50**
- QWidget * **tab_5**
- QLabel * **label_12**
- QLabel * **label_36**
- QLabel * **label_44**
- QLabel * **label_65**
- QLabel * **label_61**
- QLabel * **label_57**
- QLabel * **label_53**
- QLabel * **label_49**
- QWidget * **widget_16**
- QLineEdit * **tsf_hex2DeltaLineEdit**
- QLabel * **label_75**
- QLabel * **label_76**
- QLabel * **label_77**
- QPushButton * **tsf_plusHex2PushButton**
- QPushButton * **tsf_minusHex2PushButton**
- QWidget * **widget_14**
- QLineEdit * **tsf_xyDeltaLineEdit**
- QLabel * **label_67**
- QLabel * **label_68**
- QLabel * **label_69**
- QLabel * **label_70**
- QLabel * **label_71**
- QPushButton * **tsf_plusYPushButton**
- QPushButton * **tsf_minusYPushButton**
- QPushButton * **tsf_minusXPushButton**
- QPushButton * **tsf_plusXPushButton**
- QWidget * **widget_8**
- QLineEdit * **tsf_rhoDeltaLineEdit**
- QLabel * **label_45**
- QLabel * **label_46**
- QLabel * **label_47**
- QPushButton * **tsf_plusRhoPushButton**
- QPushButton * **tsf_minusRhoPushButton**
- QWidget * **widget_15**

- QLineEdit * **tsf_hex1DeltaLineEdit**
- QLabel * **label_72**
- QLabel * **label_73**
- QLabel * **label_74**
- QPushButton * **tsf_plusHex1PushButton**
- QPushButton * **tsf_minusHex1PushButton**
- QWidget * **widget_17**
- QLineEdit * **tsf_hex3DeltaLineEdit**
- QLabel * **label_78**
- QLabel * **label_79**
- QLabel * **label_80**
- QPushButton * **tsf_plusHex3PushButton**
- QPushButton * **tsf_minusHex3PushButton**
- QWidget * **widget_18**
- QLineEdit * **tsf_hex4DeltaLineEdit**
- QLabel * **label_81**
- QLabel * **label_82**
- QLabel * **label_83**
- QPushButton * **tsf_plusHex4PushButton**
- QPushButton * **tsf_minusHex4PushButton**
- QWidget * **widget_19**
- QLineEdit * **tsf_hex5DeltaLineEdit**
- QLabel * **label_84**
- QLabel * **label_85**
- QLabel * **label_86**
- QPushButton * **tsf_plusHex5PushButton**
- QPushButton * **tsf_minusHex5PushButton**
- QWidget * **widget_20**
- QLineEdit * **tsf_hex6DeltaLineEdit**
- QLabel * **label_102**
- QLabel * **label_103**
- QLabel * **label_104**
- QPushButton * **tsf_plusHex6PushButton**
- QPushButton * **tsf_minusHex6PushButton**
- QLabel * **rel_x_tsf_mmLabel**
- QLabel * **rel_rho_tsf_degLabel**
- QLabel * **label_108**
- QLabel * **label_109**
- QLabel * **rel_y_tsf_mmLabel**
- QLabel * **rel_hex1_tsf_mmLabel**
- QLabel * **rel_hex2_tsf_mmLabel**
- QLabel * **rel_hex3_tsf_mmLabel**
- QLabel * **rel_hex6_tsf_mmLabel**
- QLabel * **rel_hex5_tsf_mmLabel**
- QLabel * **rel_hex4_tsf_mmLabel**
- QWidget * **tab_10**
- QLabel * **label_187**
- QWidget * **widget_32**
- QLineEdit * **ang_xyDeltaLineEdit**
- QLabel * **label_188**

- QLabel * **label_189**
- QLabel * **label_190**
- QLabel * **label_191**
- QLabel * **label_192**
- QPushButton * **ang_plusYPushButton**
- QPushButton * **ang_minusYPushButton**
- QPushButton * **ang_minusXPushButton**
- QPushButton * **ang_plusXPushButton**
- QLabel * **rel_y_ang_asecLabel**
- QLabel * **rel_x_ang_asecLabel**
- QLabel * **label_193**
- QLabel * **label_194**
- QWidget * **tab_3**
- QTabWidget * **abs_CoordTypeTabWidget**
- QWidget * **tab_8**
- QPushButton * **green_itfGoToPositionPushButton**
- QLabel * **label_107**
- QLabel * **abs_itf_z_mmLabel**
- QLineEdit * **itf_z_mmLineEdit**
- QPushButton * **itfClearValuesPushButton**
- QLabel * **label_10**
- QLineEdit * **itf_y_mmLineEdit**
- QLineEdit * **itf_rho_degLineEdit**
- QLabel * **abs_itf_y_mmLabel**
- QLineEdit * **itf_theta_degLineEdit**
- QLineEdit * **itf_phi_degLineEdit**
- QLabel * **label_13**
- QLabel * **label_14**
- QLineEdit * **itf_x_mmLineEdit**
- QLabel * **label_110**
- QLabel * **label_111**
- QLabel * **label_112**
- QLabel * **abs_itf_phi_degLabel**
- QLabel * **label_113**
- QLabel * **abs_itf_x_mmLabel**
- QLabel * **abs_itf_rho_degLabel**
- QLabel * **label_15**
- QLabel * **label_114**
- QLabel * **abs_itf_theta_degLabel**
- QWidget * **tab_6**
- QLabel * **label_87**
- QLineEdit * **rtf_x_mmLineEdit**
- QLabel * **label_88**
- QLineEdit * **rtf_y_mmLineEdit**
- QLabel * **label_89**
- QLineEdit * **rtf_theta_degLineEdit**
- QLabel * **label_90**
- QLineEdit * **rtf_z_mmLineEdit**
- QLabel * **label_91**
- QLineEdit * **rtf_phi_degLineEdit**

- QLabel * **label_92**
- QLineEdit * **rtf_rho_degLineEdit**
- QPushButton * **green_rtfGoToPositionPushButton**
- QPushButton * **rtfClearValuesPushButton**
- QLabel * **abs_rtf_rho_degLabel**
- QLabel * **abs_rtf_z_mmLabel**
- QLabel * **abs_rtf_theta_degLabel**
- QLabel * **abs_rtf_phi_degLabel**
- QLabel * **abs_rtf_x_mmLabel**
- QLabel * **abs_rtf_y_mmLabel**
- QLabel * **label_2**
- QLabel * **label_3**
- QLabel * **label_4**
- QLabel * **label_5**
- QWidget * **tab_7**
- QLineEdit * **tsf_y_mmLineEdit**
- QLineEdit * **tsf_rho_degLineEdit**
- QLabel * **label_93**
- QLineEdit * **tsf_x_mmLineEdit**
- QLabel * **label_94**
- QLabel * **label_95**
- QLabel * **label_96**
- QLineEdit * **tsf_hex1_mmLineEdit**
- QLabel * **label_97**
- QLineEdit * **tsf_hex2_mmLineEdit**
- QLabel * **label_98**
- QLineEdit * **tsf_hex3_mmLineEdit**
- QLabel * **label_99**
- QLineEdit * **tsf_hex4_mmLineEdit**
- QLabel * **label_100**
- QLineEdit * **tsf_hex6_mmLineEdit**
- QLabel * **label_101**
- QLineEdit * **tsf_hex5_mmLineEdit**
- QPushButton * **green_tsfGoToPositionPushButton**
- QPushButton * **tsfClearValuesPushButton**
- QLabel * **abs_hex1_tsf_mmLabel**
- QLabel * **abs_hex2_tsf_mmLabel**
- QLabel * **abs_hex3_tsf_mmLabel**
- QLabel * **abs_hex4_tsf_mmLabel**
- QLabel * **abs_y_tsf_mmLabel**
- QLabel * **abs_rho_tsf_degLabel**
- QLabel * **label_7**
- QLabel * **label_8**
- QLabel * **label_6**
- QLabel * **label_9**
- QLabel * **abs_hex5_tsf_mmLabel**
- QLabel * **abs_hex6_tsf_mmLabel**
- QLabel * **abs_ux_tsf_mmLabel**
- QLabel * **abs_lx_tsf_mmLabel**
- QWidget * **tab_11**

- QLabel * **label_136**
- QLineEdit * **ang_y_asecLineEdit**
- QLabel * **label_64**
- QLabel * **label_137**
- QLabel * **label_138**
- QLineEdit * **ang_x_asecLineEdit**
- QLabel * **abs_x_ang_asecLabel**
- QPushButton * **green_angGoToPositionPushButton**
- QLabel * **abs_y_ang_asecLabel**
- QPushButton * **angClearValuesPushButton**
- QWidget * **tab_12**
- QWidget * **widget_9**
- QPushButton * **upperCenterPresetsPushButton**
- QPushButton * **homePresetsPushButton**
- QPushButton * **leftPresetsPushButton**
- QPushButton * **rightPresetsPushButton**
- QPushButton * **upperLeftPresetsPushButton**
- QPushButton * **upperRightPresetsPushButton**
- QPushButton * **lowerRightPresetsPushButton**
- QPushButton * **lowerLeftPresetsPushButton**
- QPushButton * **centerPresetsPushButton**
- QGroupBox * **groupBox_3**
- QGridLayout * **gridLayout_3**
- QLabel * **label_11**
- QLabel * **label_66**
- QRadioButton * **velocitySlowSlewRadioButton**
- QRadioButton * **velocitySlowTrackRadioButton**
- QRadioButton * **velocityFastSlewRadioButton**
- QRadioButton * **velocityFastTrackRadioButton**
- QGroupBox * **groupBox**
- QGridLayout * **gridLayout**
- QLedIndicator * **x_motionLedIndicator**
- QLedIndicator * **y_motionLedIndicator**
- QLedIndicator * **rho_motionLedIndicator**
- QLedIndicator * **hex1_motionLedIndicator**
- QLedIndicator * **hex2_motionLedIndicator**
- QLedIndicator * **hex3_motionLedIndicator**
- QLedIndicator * **hex4_motionLedIndicator**
- QLedIndicator * **hex5_motionLedIndicator**
- QLedIndicator * **hex6_motionLedIndicator**
- QPushButton * **red_abortTrackerPushButton**
- QButtonGroup * **angButtonGroup**
- QButtonGroup * **tsfButtonGroup**
- QButtonGroup * **velocityButtonGroup**
- QButtonGroup * **rtfButtonGroup**
- QButtonGroup * **presetsButtonGroup**
- QButtonGroup * **itfButtonGroup**

The documentation for this class was generated from the following file:

- gui/build/ui_trackerTab.h

9.1022 Ui_trajectoryTab Class Reference

Public Member Functions

- void **setupUi** (QWidget *trajectoryTab)
- void **retranslateUi** (QWidget *trajectoryTab)

Public Attributes

- QGridLayout * **gridLayout_7**
- QSpacerItem * **horizontalSpacer_6**
- QSpacerItem * **horizontalSpacer_4**
- QPushButton * **red_abortTrackerPushButton**
- QTcsFrame * **tcsFrame_2**
- QGridLayout * **gridLayout_6**
- QLabel * **label_15**
- QSpacerItem * **horizontalSpacer**
- QPushButton * **green_loadPushButton**
- QTabWidget * **trajectoryTabWidget**
- QWidget * **tab_9**
- QLabel * **label_70**
- QLineEdit * **raDecRaLineEdit**
- QLineEdit * **raDecDecLineEdit**
- QLabel * **label_69**
- QLineEdit * **raDecAzLineEdit**
- QLabel * **label_87**
- QGroupBox * **groupBox**
- QGridLayout * **gridLayout**
- QRadioButton * **raDecEastRadioButton**
- QRadioButton * **raDecWestRadioButton**
- QLineEdit * **raDecEquinoxLineEdit**
- QLabel * **label_91**
- QLineEdit * **raDecEpochLineEdit**
- QLabel * **label_92**
- QGroupBox * **groupBox_4**
- QPushButton * **raDecEastTrajectoryPushButton**
- QPushButton * **raDecSouthTrajectoryPushButton**
- QPushButton * **raDecWestTrajectoryPushButton**
- QWidget * **tab_10**
- QLabel * **label_71**
- QLineEdit * **xyYLineEdit**
- QLineEdit * **xyXLineEdit**
- QLabel * **label_72**
- QLineEdit * **xyAzLineEdit**
- QLabel * **label_94**
- QWidget * **tab_15**
- QLineEdit * **fileFileLineEdit**
- QLabel * **label_74**
- QLineEdit * **fileRaLineEdit**
- QLineEdit * **fileDecLineEdit**

- QLabel * **label_73**
- QLabel * **label_76**
- QPushButton * **fileFileBrowsePushButton**
- QWidget * **tab**
- QLabel * **label_75**
- QLineEdit * **satFileLineEdit**
- QLabel * **label_88**
- QLineEdit * **satObjectLineEdit**
- QLabel * **label_78**
- QLineEdit * **satAzLineEdit**
- QPushButton * **satFileBrowsePushButton**
- QWidget * **tab_2**
- QLabel * **label_89**
- QLabel * **label_77**
- QLineEdit * **ephemFileLineEdit**
- QGroupBox * **groupBox_2**
- QGridLayout * **gridLayout_2**
- QRadioButton * **ephemEastRadioButton**
- QRadioButton * **ephemWestRadioButton**
- QLineEdit * **ephemAzLineEdit**
- QPushButton * **ephemFileBrowsePushButton**
- QGroupBox * **groupBox_1**
- QGridLayout * **gridLayout_5**
- QGroupBox * **groupBox_3**
- QGridLayout * **gridLayout_3**
- QCheckBox * **moveDomeCheckBox**
- QCheckBox * **moveStructureCheckBox**
- QCheckBox * **moveProbesCheckBox**
- QCheckBox * **correctParallacticCheckBox**
- QPushButton * **yellow_goNextPushButton**
- QPushButton * **correctForStructurePushButton**
- QPushButton * **correctForParallacticPushButton**
- QSpacerItem * **verticalSpacer_2**
- QPushButton * **red_cancelPushButton**
- QTcsFrame * **tcsFrame_3**
- QGridLayout * **gridLayout_4**
- QLabel * **currentNumPointsLabel**
- QLabel * **nextDecLabel**
- QLabel * **currentAzimuthLabel**
- QLabel * **label_12**
- QLabel * **label_17**
- QLabel * **nextRaLabel**
- QLabel * **nextDurationLabel**
- QLabel * **currentDecLabel**
- QLabel * **label_11**
- QLabel * **label_10**
- QLabel * **nextAzimuthLabel**
- QLabel * **label_14**
- QLabel * **label_7**
- QLabel * **currentTrackStartTimeLabel**

- QLabel * **currentDurationLabel**
- QLabel * **currentRaLabel**
- QLabel * **nextTrackStartTimeLabel**
- QLabel * **label_13**
- QLabel * **currentAltitudeLabel**
- QLabel * **label_9**
- QLabel * **label_8**
- QLabel * **nextNumPointsLabel**
- QLabel * **label_6**
- QTextEdit * **trackProgressTextEdit**
- QLedIndicator * **trajectoryLedIndicator**
- QLabel * **label_19**
- QLabel * **label_5**
- QLabel * **nextAltitudeLabel**
- QLabel * **trackTimeRemainingLabel**
- QLabel * **trackTimeUntilStartLabel**
- QLabel * **trackStartTimeLabel**
- QLabel * **trackStopTimeLabel**
- QFrame * **line**
- QLabel * **label_18**
- QTcsFrame * **tcsFrame**
- QTabWidget * **offsetTypeTabWidget**
- QWidget * **tab_4**
- QLabel * **label_177**
- QWidget * **widget_28**
- QLineEdit * **tipTilt_thetaPhiDeltaLineEdit**
- QLabel * **label_165**
- QLabel * **label_166**
- QLabel * **label_167**
- QLabel * **label_168**
- QLabel * **label_169**
- QPushButton * **tipTilt_plusThetaPushButton**
- QPushButton * **tipTilt_minusThetaPushButton**
- QPushButton * **tipTilt_minusPhiPushButton**
- QPushButton * **tipTilt_plusPhiPushButton**
- QWidget * **tab_3**
- QLabel * **label**
- QWidget * **widget_7**
- QLineEdit * **RaDecDeltaLineEdit**
- QLabel * **label_31**
- QLabel * **label_33**
- QLabel * **label_34**
- QLabel * **label_37**
- QLabel * **label_43**
- QPushButton * **plusDecPushButton**
- QPushButton * **minusDecPushButton**
- QPushButton * **minusRaPushButton**
- QPushButton * **plusRaPushButton**
- QLabel * **RaOffsetLabel**
- QLabel * **label_106**

- QLabel * **label_105**
- QLabel * **DecOffsetLabel**
- QPushButton * **resetRaDecCumulativeOffsetPushButton**
- QCheckBox * **raDecCorrectionCheckBox**
- QWidget * **tab_5**
- QWidget * **widget_9**
- QLineEdit * **xyDeltaLineEdit**
- QLabel * **label_35**
- QLabel * **label_38**
- QLabel * **label_39**
- QLabel * **label_40**
- QLabel * **label_45**
- QPushButton * **plusYPushButton**
- QPushButton * **minusYPushButton**
- QPushButton * **minusXPushButton**
- QPushButton * **plusXPushButton**
- QLabel * **label_3**
- QCheckBox * **xyCorrectionCheckBox**
- QWidget * **tab_6**
- QWidget * **widget_10**
- QLineEdit * **rhoDeltaLineEdit**
- QLabel * **label_41**
- QLabel * **label_42**
- QLabel * **label_46**
- QPushButton * **plusRhoPushButton**
- QPushButton * **minusRhoPushButton**
- QLabel * **label_4**
- QWidget * **widget_8**
- QLineEdit * **focusDeltaLineEdit**
- QLabel * **label_32**
- QLabel * **label_36**
- QLabel * **label_44**
- QPushButton * **plusFocusPushButton**
- QPushButton * **minusFocusPushButton**
- QLabel * **label_2**
- QLabel * **label_16**
- QButtonGroup * **raDecDirectionButtonGroup**
- QButtonGroup * **raDecButtonGroup**
- QButtonGroup * **ephemDirectionButtonGroup**
- QButtonGroup * **rhoButtonGroup**
- QButtonGroup * **focusButtonGroup**
- QButtonGroup * **tipTiltButtonGroup**
- QButtonGroup * **xyButtonGroup**

The documentation for this class was generated from the following file:

- gui/build/ui_trajectoryTab.h

9.1023 Ui_videoSwitchTab Class Reference

Public Member Functions

- void **setupUi** (QMainWindow ∗videoSwitchTab)
- void **retranslateUi** (QMainWindow ∗videoSwitchTab)

Public Attributes

- QAction ∗ **viewNumbersAction**
- QAction ∗ **viewLabelsAction**
- QAction ∗ **viewMenuBarAction**
- QAction ∗ **viewStatusAction**
- QWidget ∗ **centralwidget**
- QGridLayout ∗ **videoSwitchTabGridLayout**
- QMenuBar ∗ **videoSwitchTabMenuBar**
- QMenu ∗ **viewMenu**

The documentation for this class was generated from the following file:

- gui/build/ui_videoSwitchTab.h

9.1024 Ui_virusEnvironmentTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗virusEnvironmentTab)
- void **retranslateUi** (QWidget ∗virusEnvironmentTab)

Public Attributes

- QGridLayout ∗ **gridLayout_7**
- QTabWidget ∗ **virusEnvTabWidget**
- QWidget ∗ **virusEnclWidget**
- QTcsFrame ∗ **tcsFrame**
- QGridLayout ∗ **gridLayout_4**
- QLabel ∗ **label**
- QGroupBox ∗ **groupBox_24**
- QGridLayout ∗ **gridLayout**
- QLedIndicator ∗ **encl1EENCTempAlarmLedIndicator**
- QLedIndicator ∗ **encl1OverwriteAlarmLedIndicator**
- QLabel ∗ **label_8**
- QLabel ∗ **label_4**
- QLedIndicator ∗ **encl1VENCTempAlarmLedIndicator**
- QLedIndicator ∗ **encl1VENCHeatExchangerInletTempAlarmLedIndicator**
- QLabel ∗ **label_110**
- QLabel ∗ **label_108**
- QLedIndicator ∗ **encl1VENCHeatExchangerOutletTempAlarmLedIndicator**
- QLabel ∗ **label_115**
- QSparkLine ∗ **encl1LowerEENCTempSparkLine**
- QLabel ∗ **label_9**
- QSparkLine ∗ **encl1VENCHeatExchangerInletTempSparkLine**
- QLabel ∗ **encl1PurgeAirTempLabel**
- QLabel ∗ **encl1PurgeAirDewPointTempLabel**
- QLabel ∗ **label_2**
- QSparkLine ∗ **encl1PurgeAirFlowSparkLine**
- QSparkLine ∗ **encl1PurgeAirDewPointTempSparkLine**
- QSparkLine ∗ **encl1VENCHeatExchangerOutletTempSparkLine**
- QLabel ∗ **label_107**
- QLabel ∗ **encl1VENCPressureLabel**
- QSparkLine ∗ **encl1FarUpperVENCTempSparkLine**
- QLabel ∗ **label_117**
- QLabel ∗ **encl1VENCHeatExchangerOutletTempLabel**
- QLedIndicator ∗ **encl1MidEENCFluidAlarmLedIndicator**
- QLabel ∗ **encl1PurgeAirFlowLabel**
- QLabel ∗ **label_111**
- QLabel ∗ **label_114**
- QLabel ∗ **encl1FarLowerVENCTempLabel**
- QLabel ∗ **encl1FarUpperVENCTempLabel**
- QLabel ∗ **encl1LowerEENCTempLabel**
- QLabel ∗ **label_113**
- QSparkLine ∗ **encl1VENCPressureSparkLine**

- QSparkLine * **encl1FarLowerVENCTempSparkLine**
- QLabel * **label_118**
- QSparkLine * **encl1HEPAPressureSparkLine**
- QSparkLine * **encl1PurgeAirTempSparkLine**
- QLabel * **encl1VENCHeatExchangerInletTempLabel**
- QLabel * **label_112**
- QLedIndicator * **encl1LowerEENCFluidAlarmLedIndicator**
- QLabel * **label_5**
- QLabel * **label_109**
- QLedIndicator * **encl1VENCOtherAlarmsLedIndicator**
- QLabel * **encl1HEPAPressureLabel**
- QLabel * **label_116**
- QLedIndicator * **encl1PurgeAirTempAlarmLedIndicator**
- QLedIndicator * **encl1PurgeAirValueLedIndicator**
- QGroupBox * **groupBox_25**
- QGridLayout * **gridLayout_2**
- QLabel * **encl2PurgeAirTempLabel**
- QSparkLine * **encl2PurgeAirFlowSparkLine**
- QLabel * **label_6**
- QLedIndicator * **encl2VENCTempAlarmLedIndicator**
- QLabel * **label_130**
- QLabel * **label_127**
- QLabel * **encl2HEPAPressureLabel**
- QLabel * **encl2PurgeAirFlowLabel**
- QSparkLine * **encl2VENCPressureSparkLine**
- QLabel * **label_126**
- QSparkLine * **encl2FarLowerVENCTempSparkLine**
- QLabel * **label_123**
- QLedIndicator * **encl2EENCTempAlarmLedIndicator**
- QLabel * **label_19**
- QLabel * **label_3**
- QLabel * **label_128**
- QLabel * **encl2FarLowerVENCTempLabel**
- QLabel * **label_125**
- QLabel * **label_121**
- QLabel * **encl2PurgeAirDewPointTempLabel**
- QLabel * **encl2VENCPressureLabel**
- QLabel * **encl2VENCHeatExchangerInletTempLabel**
- QSparkLine * **encl2PurgeAirTempSparkLine**
- QSparkLine * **encl2HEPAPressureSparkLine**
- QSparkLine * **encl2VENCHeatExchangerInletTempSparkLine**
- QLabel * **label_122**
- QLabel * **label_124**
- QLabel * **label_119**
- QLabel * **label_12**
- QSparkLine * **encl2PurgeAirDewPointTempSparkLine**
- QLedIndicator * **encl2VENCHeatExchangerInletTempAlarmLedIndicator**
- QSparkLine * **encl2VENCHeatExchangerOutletTempSparkLine**
- QLedIndicator * **encl2VENCHeatExchangerOutletTempAlarmLedIndicator**
- QLabel * **encl2LowerEENCTempLabel**

- QLabel * **encl2VENCHeatExchangerOutletTempLabel**
- QSparkLine * **encl2FarUpperVENCTempSparkLine**
- QSparkLine * **encl2LowerEENCTempSparkLine**
- QLabel * **label_120**
- QLabel * **encl2FarUpperVENCTempLabel**
- QLedIndicator * **encl2VENCOtherAlarmsLedIndicator**
- QLabel * **label_129**
- QLedIndicator * **encl2PurgeAirTempAlarmLedIndicator**
- QLedIndicator * **encl2OverwriteAlarmLedIndicator**
- QLedIndicator * **encl2LowerEENCFluidAlarmLedIndicator**
- QLedIndicator * **encl2MidEENCFluidAlarmLedIndicator**
- QLabel * **label_7**
- QLedIndicator * **encl2PurgeAirValueLedIndicator**
- QWidget * **tab**
- QTcsFrame * **tcsFrame_3**
- QLabel * **label_41**
- QGroupBox * **groupBox_27**
- QGridLayout * **gridLayout_15**
- QLabel * **label_38**
- QLabel * **label_40**
- QLabel * **label_150**
- QLabel * **encl2DomeTempLabel**
- QSparkLine * **encl2DomeTempSparkLine**
- QLabel * **label_149**
- QLabel * **encl2DomeTempAverageLabel**
- QSparkLine * **encl2DomeTempAverageSparkLine**
- QLabel * **label_151**
- QLabel * **encl2GlycolSupplyTempLabel**
- QSparkLine * **encl2GlycolSupplyTempSparkLine**
- QLabel * **label_146**
- QLabel * **encl2GlycolReturnTempLabel**
- QSparkLine * **encl2GlycolReturnTempSparkLine**
- QLabel * **label_160**
- QLabel * **encl2GlycolSupplyTempAverageLabel**
- QSparkLine * **encl2GlycolSupplyTempAverageSparkLine**
- QLabel * **label_157**
- QLabel * **encl2GlycolReturnTempAverageLabel**
- QSparkLine * **encl2GlycolReturnTempAverageSparkLine**
- QLabel * **label_153**
- QLabel * **encl2ComputedSetPointLabel**
- QSparkLine * **encl2ComputedSetPointSparkLine**
- QLabel * **label_36**
- QLabel * **encl2ThermoCubeSetPointLabel**
- QSparkLine * **encl2ThermoCubeSetPointSparkLine**
- QLabel * **label_156**
- QLabel * **encl2ThermoCubeTempLabel**
- QSparkLine * **encl1ThermoCubeTempSparkLine**
- QLabel * **label_39**
- QLedIndicator * **encl2RunningLedIndicator**
- QLabel * **label_147**

- QLedIndicator * **encl2FanFailureAlarmLedIndicator**
- QLabel * **label_154**
- QLedIndicator * **encl2PumpFailureAlarmLedIndicator**
- QLabel * **label_152**
- QLedIndicator * **encl2RTDOpenAlarmLedIndicator**
- QLabel * **label_158**
- QLedIndicator * **encl2RTDShortAlarmLedIndicator**
- QLabel * **label_159**
- QLedIndicator * **encl2TankLevelLowAlarmLedIndicator**
- QLabel * **label_148**
- QLedIndicator * **encl2NoFaultsAlarmLedIndicator**
- QLabel * **label_37**
- QLedIndicator * **encl2ThermoCubeIsCoolingLedIndicator**
- QLabel * **label_155**
- QLedIndicator * **encl2ControllingTempsLedIndicator**
- QGroupBox * **groupBox_26**
- QGridLayout * **gridLayout_14**
- QLabel * **encl1ThermoCubeTempLabel**
- QLabel * **label_30**
- QLedIndicator * **encl1TankLevelLowAlarmLedIndicator**
- QLabel * **label_131**
- QLabel * **label_133**
- QLabel * **label_32**
- QLabel * **label_33**
- QLabel * **label_134**
- QLabel * **encl1DomeTempLabel**
- QLabel * **label_35**
- QSparkLine * **encl1DomeTempAverageSparkLine**
- QSparkLine * **encl1DomeTempSparkLine**
- QLabel * **encl1DomeTempAverageLabel**
- QLabel * **label_135**
- QLabel * **encl1ThermoCubeSetPointLabel**
- QLabel * **label_136**
- QSparkLine * **encl2ThermoCubeTempSparkLine**
- QLabel * **encl1GlycolSupplyTempLabel**
- QLabel * **label_137**
- QLabel * **label_138**
- QLedIndicator * **encl1PumpFailureAlarmLedIndicator**
- QLabel * **label_139**
- QLabel * **label_140**
- QLedIndicator * **encl1FanFailureAlarmLedIndicator**
- QSparkLine * **encl1GlycolReturnTempSparkLine**
- QSparkLine * **encl1ComputedSetPointSparkLine**
- QLabel * **encl1GlycolReturnTempAverageLabel**
- QLabel * **encl1ComputedSetPointLabel**
- QLabel * **label_141**
- QSparkLine * **encl1ThermoCubeSetPointSparkLine**
- QLabel * **label_142**
- QLabel * **encl1GlycolReturnTempLabel**
- QLedIndicator * **encl1RTDShortAlarmLedIndicator**

- QLabel * **label_143**
- QLedIndicator * **encl1RTDOpenAlarmLedIndicator**
- QSparkLine * **encl1GlycolSupplyTempSparkLine**
- QLabel * **label_144**
- QSparkLine * **encl1GlycolReturnTempAverageSparkLine**
- QLabel * **label_145**
- QLabel * **encl1GlycolSupplyTempAverageLabel**
- QSparkLine * **encl1GlycolSupplyTempAverageSparkLine**
- QLedIndicator * **encl1ThermoCubeIsCoolingLedIndicator**
- QLedIndicator * **encl1ControllingTempsLedIndicator**
- QLabel * **label_34**
- QLedIndicator * **encl1RunningLedIndicator**
- QLedIndicator * **encl1NoFaultsAlarmLedIndicator**
- QLabel * **label_132**
- QWidget * **virusTempsWidget**
- QGridLayout * **gridLayout_5**
- QLabel * **virusLastUpdateTimeLabel**
- QLabel * **virusTempsUpdateCountLabel**
- QSpacerItem * **horizontalSpacer**
- QLabel * **label_10**
- QLabel * **label_15**
- QPushButton * **virusUpdateTempsPushButton**
- QTableWidget * **virusTempsTableWidget**
- QLabel * **label_27**
- QLabel * **virusUnitsLabel**
- QWidget * **lrs2TempsWidget**
- QGridLayout * **gridLayout_3**
- QLabel * **label_28**
- QLabel * **label_11**
- QSpacerItem * **horizontalSpacer_2**
- QPushButton * **lrs2UpdateTempsPushButton**
- QLabel * **lrs2LastUpdateTimeLabel**
- QLabel * **lrs2TempsUpdateCountLabel**
- QTableWidget * **lrs2TempsTableWidget**
- QLabel * **label_20**
- QLabel * **lrs2UnitsLabel**
- QWidget * **warmUpCoolDownWidget**
- QTcsFrame * **tcsFrame_2**
- QGridLayout * **gridLayout_12**
- QGroupBox * **groupBox_3**
- QGridLayout * **gridLayout_11**
- QLabel * **label_22**
- QLabel * **cryoTempLabel**
- QLineEdit * **tempTargetSetpointLeftLineEdit**
- QLabel * **label_17**
- QGroupBox * **groupBox_4**
- QGridLayout * **gridLayout_9**
- QLabel * **label_24**
- QLabel * **leftCCDIdLabel**
- QLabel * **label_23**

- QLabel * **leftCCDTempLabel**
- QLabel * **leftCCDTempSetpointLabel**
- QLabel * **label_21**
- QGroupBox * **groupBox_5**
- QGridLayout * **gridLayout_10**
- QLabel * **label_31**
- QLabel * **rightCCDIdLabel**
- QLabel * **label_29**
- QLabel * **rightCCDTempLabel**
- QLabel * **label_26**
- QLabel * **rightCCDTempSetpointLabel**
- QLineEdit * **tempTargetSetpointRightLineEdit**
- QLabel * **label_13**
- QLabel * **label_14**
- QPushButton * **executeWarmUpCoolDownPushButton**
- QSpacerItem * **verticalSpacer**
- QLabel * **spacingLabel_2**
- QLabel * **spacingLabel**
- QGroupBox * **groupBox_6**
- QGridLayout * **gridLayout_13**
- QLineEdit * **specIdLineEdit**
- QLabel * **label_16**
- QLabel * **label_25**
- QLabel * **label_18**
- QLabel * **cntrlIdLabel**
- QLabel * **muxIdLabel**
- QGroupBox * **groupBox_2**
- QGridLayout * **gridLayout_8**
- QRadioButton * **warmUpRadioButton**
- QRadioButton * **coolDownRadioButton**
- QGroupBox * **groupBox**
- QGridLayout * **gridLayout_6**
- QRadioButton * **lrs2RadioButton**
- QRadioButton * **virusRadioButton**
- QSpacerItem * **verticalSpacer_3**
- QPushButton * **updateTempsPushButton**

The documentation for this class was generated from the following file:

- gui/build/ui_virusEnvironmentTab.h

9.1025 Ui_weatherDataTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗weatherDataTab)
- void **retranslateUi** (QWidget ∗weatherDataTab)

Public Attributes

- QGroupBox ∗ **groupBox_18**
- QGridLayout ∗ **gridLayout**
- QLabel ∗ **label_33**
- QLabel ∗ **currTempLabel**
- QLabel ∗ **label_27**
- QLabel ∗ **avgTempLabel**
- QLabel ∗ **label_25**
- QLabel ∗ **maxTempLabel**
- QLabel ∗ **label_36**
- QLabel ∗ **minTempLabel**
- QGroupBox ∗ **groupBox_19**
- QGridLayout ∗ **gridLayout_2**
- QLabel ∗ **label_34**
- QLabel ∗ **currHumidityLabel**
- QLabel ∗ **label_28**
- QLabel ∗ **avgHumidityLabel**
- QLabel ∗ **label_26**
- QLabel ∗ **maxHumidityLabel**
- QLabel ∗ **label_37**
- QLabel ∗ **minHumidityLabel**
- QGroupBox ∗ **groupBox_20**
- QGridLayout ∗ **gridLayout_3**
- QLabel ∗ **label_35**
- QLabel ∗ **currIntDewPointLabel**
- QLabel ∗ **label_29**
- QLabel ∗ **avgIntDewPointLabel**
- QLabel ∗ **label_30**
- QLabel ∗ **maxIntDewPointLabel**
- QLabel ∗ **label_38**
- QLabel ∗ **minIntDewPointLabel**
- QGroupBox ∗ **groupBox_21**
- QGridLayout ∗ **gridLayout_4**
- QLabel ∗ **label_39**
- QLabel ∗ **currExtDewPointLabel**
- QLabel ∗ **label_31**
- QLabel ∗ **avgExtDewPointLabel**
- QLabel ∗ **label_32**
- QLabel ∗ **maxExtDewPointLabel**
- QLabel ∗ **label_40**
- QLabel ∗ **minExtDewPointLabel**
- QGroupBox ∗ **groupBox_22**

- QGridLayout * **gridLayout_5**
- QLabel * **label_41**
- QLabel * **currPressureLabel**
- QLabel * **label_42**
- QLabel * **avgPressureLabel**
- QLabel * **label_43**
- QLabel * **maxPressureLabel**
- QLabel * **label_44**
- QLabel * **minPressureLabel**
- QGroupBox * **groupBox_23**
- QGridLayout * **gridLayout_6**
- QLabel * **label_45**
- QLabel * **currWindDirLabel**
- QLabel * **label_46**
- QLabel * **avgWindDirLabel**
- QLabel * **label_47**
- QLabel * **stdDevWindDirLabel**
- QLabel * **label_48**
- QLabel * **modeWindDirLabel**
- QGroupBox * **groupBox_24**
- QGridLayout * **gridLayout_7**
- QLabel * **label_49**
- QLabel * **currWindSpeedLabel**
- QLabel * **label_50**
- QLabel * **avgWindSpeedLabel**
- QLabel * **label_51**
- QLabel * **maxWindSpeedLabel**
- QLabel * **label_52**
- QLabel * **minWindSpeedLabel**
- QGroupBox * **groupBox_25**
- QGridLayout * **gridLayout_8**
- QLabel * **label_53**
- QLabel * **currExtDust1umLabel**
- QLabel * **label_54**
- QLabel * **avgExtDust1umLabel**
- QLabel * **label_55**
- QLabel * **maxExtDust1umLabel**
- QLabel * **label_56**
- QLabel * **minExtDust1umLabel**
- QGroupBox * **groupBox_26**
- QGridLayout * **gridLayout_9**
- QLabel * **label_57**
- QLabel * **currIntDust1umLabel**
- QLabel * **label_58**
- QLabel * **avgIntDust1umLabel**
- QLabel * **label_59**
- QLabel * **maxIntDust1umLabel**
- QLabel * **label_60**
- QLabel * **minIntDust1umLabel**
- QGroupBox * **groupBox_27**

- QGridLayout * **gridLayout_10**
- QLabel * **label_61**
- QLabel * **currRainLabel**
- QLabel * **label_62**
- QLabel * **sumRainLabel**
- QGroupBox * **groupBox_28**
- QGridLayout * **gridLayout_11**
- QLabel * **label_65**
- QLabel * **currExtDustPt3umLabel**
- QLabel * **label_66**
- QLabel * **avgExtDustPt3umLabel**
- QLabel * **label_67**
- QLabel * **maxExtDustPt3umLabel**
- QLabel * **label_68**
- QLabel * **minExtDustPt3umLabel**
- QGroupBox * **groupBox_29**
- QGridLayout * **gridLayout_12**
- QLabel * **label_69**
- QLabel * **currIntDustPt3umLabel**
- QLabel * **label_70**
- QLabel * **avgIntDustPt3umLabel**
- QLabel * **label_71**
- QLabel * **maxIntDustPt3umLabel**
- QLabel * **label_72**
- QLabel * **minIntDustPt3umLabel**
- QSparkLine * **currTempSparkLine**
- QLabel * **label**

The documentation for this class was generated from the following file:

- gui/build/ui_weatherDataTab.h

9.1026 Ui_webBrowserTab Class Reference

Public Member Functions

- void **setupUi** (QWidget ∗webBrowserTab)
- void **retranslateUi** (QWidget ∗webBrowserTab)

Public Attributes

- QGridLayout ∗ **gridLayout**
- QTcsFrame ∗ **tcsFrame**
- QGridLayout ∗ **gridLayout_2**
- QLabel ∗ **label**
- QWebView ∗ **thisWebView**
- QComboBox ∗ **urlComboBox**
- QPushButton ∗ **backPushButton**
- QPushButton ∗ **forwardPushButton**

The documentation for this class was generated from the following file:

- gui/build/ui_webBrowserTab.h

9.1027 UnhandledHandler Class Reference

Public Member Functions

- virtual void process (Message *message)

9.1027.1 Member Function Documentation

9.1027.1.1 virtual void UnhandledHandler::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- het_lib/messaging/include/builtin_handlers.h

9.1028 Unimplemented Class Reference

Public Member Functions

- void **init** (void)

The documentation for this class was generated from the following file:

- ics/ics.h

9.1029 UnknownException Class Reference

Public Member Functions

- **UnknownException** (tcsexcept::tcs_exception_t __unknown, const std::string &w, const char *file, int line, const char *func)

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_exception.h

9.1030 unset_position Class Reference

9.1030.1 Detailed Description

Unset the target offset for the probe. A subsequent move of another probe may result in the unset probe being moved.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **unset_position** (TCSDriver *d, PFIPClient &pfip, Instrument &instrument, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.1030.2 Member Function Documentation

9.1030.2.1 void unset_position::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/pfip_client.h

9.1031 update_setup Class Reference

9.1031.1 Detailed Description

Marks the system as being setup and ready for science instrument operations, or not.

Required parameters:

state<bool> - Flag indicating the state of the system.

`#include <tcs_handler_doc.h>`

Public Member Functions

- **update_setup** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.1031.2 Member Function Documentation

9.1031.2.1 void update_setup::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/tcs_handlers.h

9.1032 UptimeHandler Class Reference

Public Member Functions

- virtual void process (Message ∗message)

9.1032.1 Member Function Documentation

9.1032.1.1 virtual void UptimeHandler::process (Message ∗ *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- het_lib/messaging/include/builtin_handlers.h

9.1033 utc Class Reference

9.1033.1 Detailed Description

Structure used to store broken down time.

`#include <fptime.h>`

Public Attributes

- int **year**
- int **month**
- int **day**
- int **hour**
- int **minute**
- int **second**
- double **microsecond**

The documentation for this class was generated from the following file:

- het_lib/common/include/fptime.h

9.1034 utc_iso8601 Class Reference

Public Member Functions

- **utc_iso8601** (const char *s)
- **utc_iso8601** (const std::string &s)

The documentation for this class was generated from the following file:

- het_lib/common/include/fptime.h

9.1035 utilityGui Class Reference

Public Member Functions

- **utilityGui** (const string &appName, guiConfig *pConfig)
- virtual int **start** (void)
- void **saveSettings** (void)
- guiConfig * **getConfig** (void)
- QApplication * **getApp** (void)
- msgid_t **getLocalMsgId** (void)
- void **showSplashScreenMessage** (QString msg, int alignment=Qt::AlignTop|Qt::AlignHCenter, const QColor=Qt::white)

Public Attributes

- QString **m_loginId**
- QString **m_SettingsFilename**
- QString **m_StyleSheetFilename**
- QSettings * **m_pSettings**
- guiPreferences **m_preferences**
- QTimer * **m_pSecondTimer**
- utilityGuiThreadWorker * **m_pCommandReceiverThreadWorker**
- utilityGuiThreadWorker * **m_pRelaySignalsThreadWorker**
- utilityGuiThreadWorker * **m_pReceiveStatusEventsThreadWorker**

Protected Member Functions

- void **loadSettings** (void)
- void **CommandReceiverThreadEntryPoint** (void)
- void **RelaySignalsThreadEntryPoint** (void)
- void **ReceiveStatusEventsThreadEntryPoint** (void)

Protected Attributes

- tthread::thread * **m_pCommandReceiverThread**
- volatile bool **m_bCommandReceiverThreadRunning**
- tthread::mutex **m_CommandReceiverThreadAccessMutex**
- guiThreadState_t **m_CommandReceiverThreadState**
- string **m_CommandReceiverThreadStartupError**
- tthread::thread * **m_pRelaySignalsThread**
- volatile bool **m_bRelaySignalsThreadRunning**
- tthread::mutex **m_RelaySignalsThreadAccessMutex**
- guiThreadState_t **m_RelaySignalsThreadState**
- string **m_RelaySignalsThreadStartupError**
- QEventLoop * **m_pRelaySignalsEventLoop**
- tthread::thread * **m_pReceiveStatusEventsThread**
- volatile bool **m_bReceiveStatusEventsThreadRunning**
- tthread::mutex **m_ReceiveStatusEventsThreadAccessMutex**
- guiThreadState_t **m_ReceiveStatusEventsThreadState**

- string **m_ReceiveStatusEventsThreadStartupError**
- EventSink * **m_pReceiveStatusEventsSink**
- Receiver * **m_pCommandReceiver**
- QApplication * **m_pQtApp**
- QSplashScreen * **m_pSplashScreen**
- guiConfig * **m_pConfig**
- int **m_localMsgId**
- tthread::mutex **m_localMsgIdMutex**

Friends

- class utilityGuiThreadWorker

The documentation for this class was generated from the following files:

- utility_gui/include/utilityGui.h
- utility_gui/src/utilityGui.cpp

9.1036 utilityGuiThreadWorker Class Reference

Signals

- void **CommandIssued** (QString, QString)
- void **CommandDone** (QString, bool)
- void **CommandError** (QString, QString)

Public Member Functions

- **utilityGuiThreadWorker** (const char *name, utilityGui *pParent)
- void **CommandReceiver** (void)
- void **RelaySignals** (void)
- void **ReceiveStatusEvents** (void)

Protected Member Functions

- void **generateIdAndStartTime** (int &id, string &startTime)
- void **processErrorEvent** (const string &system, const string &source, const string &key, Event *pEvent)
- void **processTCSStatusEvent** (Event *pEvent)

Protected Attributes

- utilityGui * **m_pParent**

The documentation for this class was generated from the following files:

- utility_gui/include/utilityGui.h
- utility_gui/src/utilityGui.cpp

9.1037 validPixelFormat Struct Reference

Public Attributes

- string **name**
- PixelFormat_t **type**
- bool **bSupported**

The documentation for this struct was generated from the following file:

- pas/src/pasConfig.cpp

9.1038 wfs_photometry::WFSCentroid::vector_t Struct Reference

Public Attributes

- double **X**
- double **Y**
- double **Length**
- double **Angle**

The documentation for this struct was generated from the following file:

- het_lib/wfs_lib/include/wfs_photometry.h

9.1039 Venc1PrintRegister Class Reference

9.1039.1 Detailed Description

Print the value of a modbus register in the Virus Enclosure 1 (VENC) PLC into the PFIP server log.

Required parameters:

regnum<int> - The register number to read

```
#include <pfip_handler_doc.h>
```

The documentation for this class was generated from the following file:

- pfip/include/pfip_handler_doc.h

9.1040 Venc1WriteRegister Class Reference

9.1040.1 Detailed Description

Store an integer value into the Virus Enclosure 1 (VENC) PLC.

Required parameters:

regnum<int> - The register number to write.

value<int> - The value to store in the register.

`#include <pfip_handler_doc.h>`

The documentation for this class was generated from the following file:

- pfip/include/pfip_handler_doc.h

9.1041 Venc2PrintRegister Class Reference

9.1041.1 Detailed Description

Print the value of a modbus register in the Virus Enclosure 2 (VENC) PLC into the PFIP server log.

Required parameters:

regnum<int> - The register number to read

`#include <pfip_handler_doc.h>`

The documentation for this class was generated from the following file:

- pfip/include/pfip_handler_doc.h

9.1042 Venc2WriteRegister Class Reference

9.1042.1 Detailed Description

Store an integer value into the Virus Enclosure 2 (VENC) PLC.

Required parameters:

regnum<int> - The register number to write.

value<int> - The value to store in the register.

`#include <pfip_handler_doc.h>`

The documentation for this class was generated from the following file:

- pfip/include/pfip_handler_doc.h

9.1043 VENCHandler Class Reference

A class derived from PFIPHandler to provide message handlers for FCU commands.

`#include <pfipServer.h>`

Public Member Functions

- **VENCHandler** (const string &name, Receiver *rcv, Alarm_Monitor *mon, MessageHandlerDescriptor &usage)

Protected Attributes

- Alarm_Monitor * **m_pMonitor**

The documentation for this class was generated from the following file:

- pfip/include/pfipServer.h

9.1044 Ui::videoSwitchTab Class Reference

The documentation for this class was generated from the following file:

- gui/build/ui_videoSwitchTab.h

9.1045 videoSwitchTabUi Class Reference

Public Slots

- void **CommandError** (QString)
- void **CommandComplete** (QString, QString)

Signals

- void **SendCommand** (QString, QString)

Public Member Functions

- **videoSwitchTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Public Attributes

- videoSwitchTabUiThreadWorker * **m_pSendCommandsThreadWorker**

Protected Types

- typedef QMap< QString, QRadioButton * > **RadioButtonMap_t**

Protected Member Functions

- void **constructTab** (void)
- void **deconstructTab** ()
- void **loadSettings** (void)
- void **saveSettings** (void)
- void **messageBox** (QString msg)
- void **SendCommandsThreadEntryPoint** (void)

Protected Attributes

- QShortcut * **m_pToggleMenuBarShortcut**
- TCSSocket * **m_Socket**
- bool **m_EmptySocketBeforeNextCmd**
- tthread::mutex **m_MutexSendCmd**
- bool **m_bShowMenuBar**
- bool **m_bShowNumbers**
- bool **m_bShowLabels**
- bool **m_bShowStatus**
- RadioButtonMap_t **m_RadioButtonMap**
- QList< QString > **m_Buttons**
- int **m_NumberOfButtons**
- QMap< QString, string > **m_Commands**
- QMap< QString, string > **m_Descriptions**

- QLineEdit * **m_pStatusLine**
- QString **m_LastStatusMsg**
- tthread::thread * **m_pSendCommandsThread**
- volatile bool **m_bSendCommandsThreadRunning**
- tthread::mutex **m_SendCommandsThreadAccessMutex**
- guiThreadState_t **m_SendCommandsThreadState**
- string **m_SendCommandsThreadStartupError**
- QEventLoop * **m_pSendCommandsEventLoop**

Friends

- class videoSwitchTabUiThreadWorker

The documentation for this class was generated from the following files:

- gui/include/videoSwitchTabUi.h
- gui/build/moc_videoSwitchTabUi.cpp
- gui/src/videoSwitchTabUi.cpp

9.1046 videoSwitchTabUiThreadWorker Class Reference

Public Slots

- void **SendCommand** (QString, QString)

Signals

- void **CommandError** (QString)
- void **CommandComplete** (QString, QString)

Public Member Functions

- **videoSwitchTabUiThreadWorker** (const char *name, videoSwitchTabUi *pParent)
- void **SendCommands** (void)

Protected Member Functions

- bool **sendCmd** (char *cmdString, const char *terminators="\n")

Protected Attributes

- videoSwitchTabUi * **m_pParent**

The documentation for this class was generated from the following files:

- gui/include/videoSwitchTabUi.h
- gui/build/moc_videoSwitchTabUi.cpp
- gui/src/videoSwitchTabUi.cpp

9.1047 VIRUS_Monitor Class Reference

Public Member Functions

- **VIRUS_Monitor** (string monitorName, string enclName, int statusFrequency)
- void **checkEnvironment** ()

The documentation for this class was generated from the following files:

- pfip/include/PFIPMonitor.h
- pfip/src/PFIPMonitor.cpp

9.1048 Ui::virusEnvironmentTab Class Reference

The documentation for this class was generated from the following file:

- gui/build/ui_virusEnvironmentTab.h

9.1049 virusEnvironmentTabUi Class Reference

Classes

- struct virusTempData

Signals

- void **ForceHeartbeat** (std::string)
- void **SetVirusCCDTemp** (int, double)
- void **ForceUpdateVirusTemps** (void)
- void **SetLRS2CCDTemp** (int, double)
- void **ForceUpdateLRS2Temps** (void)

Public Member Functions

- **virusEnvironmentTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Protected Member Functions

- void **loadSettings** (void)
- void **saveSettings** (void)
- void **clearWarmUpCoolDownFields** (void)

Protected Attributes

- int **m_virusTempsRowCount**
- int **m_virusTempsUpdateCount**
- string **m_virusLastUpdateTime**
- int **m_virusUnits**
- int **m_lrs2TempsRowCount**
- int **m_lrs2TempsUpdateCount**
- string **m_lrs2LastUpdateTime**
- int **m_lrs2Units**
- int **m_tableColumnCount**
- std::map< int, virusTempData > **m_virusTempData**
- tthread::mutex **m_virusTempDataAccessMutex**
- int **m_virusCurrentWarmUpCoolDownSpecId**
- int **m_virusSelectedRow**
- std::map< int, virusTempData > **m_lrs2TempData**
- tthread::mutex **m_lrs2TempDataAccessMutex**
- int **m_lrs2CurrentWarmUpCoolDownSpecId**
- int **m_lrs2SelectedRow**

The documentation for this class was generated from the following files:

- gui/include/virusEnvironmentTabUi.h
- gui/build/moc_virusEnvironmentTabUi.cpp
- gui/src/virusEnvironmentTabUi.cpp

9.1050 VirusMonitorStatus Class Reference

Classes

- struct alarms
- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- status **m_status**
- alarms **m_alarms**

The documentation for this class was generated from the following files:

- gui/include/pfipStatusData.h
- gui/src/pfipStatusData.cpp

9.1051 virusEnvironmentTabUi::virusTempData Struct Reference

Public Attributes

- int **muxId**
- int **cntrlId**
- int **ihmp**
- int **encl**
- double **cryoTemp**
- int **leftId**
- double **leftTemp**
- double **leftTempSetpoint**
- double **leftHeaterVoltage**
- int **rightId**
- double **rightTemp**
- double **rightTempSetpoint**
- double **rightHeaterVoltage**

The documentation for this struct was generated from the following file:

- gui/include/virusEnvironmentTabUi.h

9.1052 camra_handlers::wait_for_readout Class Reference

Public Member Functions

- **wait_for_readout** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.1052.1 Member Function Documentation

9.1052.1.1 void camra_handlers::wait_for_readout::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- camra/camra_handlers.h

9.1053 wait_for_readout Class Reference

9.1053.1 Detailed Description

Returns when the readout is complete for the given exposure.

See also:

wait_for_shutter_open, wait_for_write

Required parameters:

exposure<int> - Identifier of the exposure to wait on.

observation<int> - Observation identifier for the exposure to wait on.

`#include <camra_handler_doc.h>`

The documentation for this class was generated from the following file:

- camra/camra_handler_doc.h

9.1054 wait_for_setup Class Reference

9.1054.1 Detailed Description

Returns when the system is flagged as being setup for observation.

Optional parameters:

timeout<double> - Amount of time to spend waiting for the setup condition to be true. If non-negative and the time expires prior to the setup, then an exception is thrown. If negative, the call waits indefinitely. Defaults to -1. (second)

`#include <tcs_handler_doc.h>`

Public Member Functions

- **wait_for_setup** (TCSDriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.1054.2 Member Function Documentation

9.1054.2.1 void wait_for_setup::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- tcs/tcs_handlers.h

9.1055 camra_handlers::wait_for_shutter_close Class Reference

Public Member Functions

- **wait_for_shutter_close** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.1055.1 Member Function Documentation

9.1055.1.1 void camra_handlers::wait_for_shutter_close::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- camra/camra_handlers.h

9.1056 wait_for_shutter_close Class Reference

9.1056.1 Detailed Description

Returns when the shutter is closed for the given exposure.

See also:

wait_for_readout, wait_for_write

Required parameters:

exposure<int> - Identifier of the exposure to wait on.

observation<int> - Observation identifier for the exposure to wait on.

`#include <camra_handler_doc.h>`

The documentation for this class was generated from the following file:

- camra/camra_handler_doc.h

9.1057 camra_handlers::wait_for_write Class Reference

Public Member Functions

- **wait_for_write** (CAMRADriver *d, const MessageHandlerDescriptor &m)
- void process (Message *message)

9.1057.1 Member Function Documentation

9.1057.1.1 void camra_handlers::wait_for_write::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements MessageHandler.

The documentation for this class was generated from the following file:

- camra/camra_handlers.h

9.1058 wait_for_write Class Reference

9.1058.1 Detailed Description

Returns when the write is complete for the given exposure.

See also:

wait_for_shutter_open, wait_for_write

Required parameters:

exposure<int> - Identifier of the exposure to wait on.

observation<int> - Observation identifier for the exposure to wait on.

`#include <camra_handler_doc.h>`

The documentation for this class was generated from the following file:

- camra/camra_handler_doc.h

9.1059 weather Class Reference

Public Member Functions

- **weather** (double pressure, double temperature, double humidity)

Public Attributes

- double **pressure_**
- double temperature_

  pressure millibars

- double relhum_

  degrees Kelvin

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_coordinates.h

9.1060 weather_GetData Class Reference

9.1060.1 Detailed Description

Optional parameters:

update_state<string> -

`#include <legacy_handler_doc.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.1061 weather_OnLine Class Reference

The documentation for this class was generated from the following file:

- legacy/include/legacy_handler_doc.h

9.1062 WeatherClient Class Reference

Public Member Functions

- **WeatherClient** (const std::string &route)
- void **update_state** (void)
- void **write** (FitsOutWrapper &f)
- **WeatherClient** (const std::string &event_route, const std::string &name)
- void **write** (FitsOutWrapper &fitsOut)
- **WeatherClient** (const std::string &route)
- weather **current** (bool ignore_events=0)
- void **update_status** (Event &event)

The documentation for this class was generated from the following files:

- camra/weather_client.h
- pas/include/weather_client.h
- tcs/weather_client.h
- pas/src/weather_client.cpp

9.1063 WeatherConfig Class Reference

Public Member Functions

- virtual void set_options (OptionParser &optionParser)

  Adds options to default set.

9.1063.1 Member Function Documentation

9.1063.1.1 void WeatherConfig::set_options (OptionParser & *optionParser*) [virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Reimplemented from SubSystemConfig.

The documentation for this class was generated from the following files:

- legacy/include/legacyConfig.h
- legacy/src/legacyConfig.cpp

9.1064 WeatherData Class Reference

Classes

- struct statistic
- struct status

Public Member Functions

- bool **initialize** (Payload &payload, void *pPrevStatus=NULL)

Public Attributes

- struct WeatherData::status **m_status**

The documentation for this class was generated from the following files:

- gui/include/legacyStatusData.h
- gui/src/legacyStatusData.cpp

9.1065 Ui::weatherDataTab Class Reference

The documentation for this class was generated from the following file:

- gui/build/ui_weatherDataTab.h

9.1066 weatherDataTabUi Class Reference

Public Member Functions

- **weatherDataTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Protected Slots

- void **UpdateWeatherData** (void)

Protected Member Functions

- virtual void **updateUi** (void)

The documentation for this class was generated from the following files:

- gui/include/weatherDataTabUi.h
- gui/src/weatherDataTabUi.cpp

9.1067 WeatherGetDataHandlerClass Class Reference

Public Member Functions

- **WeatherGetDataHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.1067.1 Member Function Documentation

9.1067.1.1 virtual void WeatherGetDataHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

The documentation for this class was generated from the following file:

- legacy/include/legacyWeatherSubSystem.h

9.1068 WeatherOnLine Class Reference

Check to see if the Weather legacy server is connected.

9.1068.1 Detailed Description

Parameters:

none

`#include <legacyWeatherSubSystem.h>`

The documentation for this class was generated from the following file:

- legacy/include/legacyWeatherSubSystem.h

9.1069 WeatherOnLineHandlerClass Class Reference

Public Member Functions

- **WeatherOnLineHandlerClass** (SubSystem *subSystem, const MessageHandlerDescriptor &d)
- virtual void process (Message *pMessage)

9.1069.1 Member Function Documentation

9.1069.1.1 virtual void WeatherOnLineHandlerClass::process (Message * *message*) [inline, virtual]

The entry point into the handler object from the perspective of the remote Handler caller. Application code inherits the Handler object, defining a process() callback with access to the parameterization of the invocation, the reply interfaces, and the application's sub-class instance.

The implementation must delete the message when it is no longer needed. Failure to delete the message object at the end of the MessageHandler process can result in a "handler is busy" error message on subsequent invocations.

The `process()` implementation should invoke the MESSAGE_SETJMP macro, see Remote signalling.

Implements SubSystemHandler.

The documentation for this class was generated from the following file:

- legacy/include/legacyWeatherSubSystem.h

9.1070 Ui::webBrowserTab Class Reference

The documentation for this class was generated from the following file:

- gui/build/ui_webBrowserTab.h

9.1071 webBrowserTabUi Class Reference

Public Member Functions

- **webBrowserTabUi** (const char *name, QWidget *pLoadedWidget, guiConfig *pConfig)

Protected Member Functions

- void **loadSettings** (void)
- void **saveSettings** (void)
- void **loadURL** (void)

Friends

- class webBrowserTabUiThreadWorker

The documentation for this class was generated from the following files:

- gui/include/webBrowserTabUi.h
- gui/src/webBrowserTabUi.cpp

9.1072 webBrowserTabUiThreadWorker Class Reference

Public Member Functions

- **webBrowserTabUiThreadWorker** (const char *, webBrowserTabUi *pParent)

Protected Attributes

- webBrowserTabUi * **m_pParent**

The documentation for this class was generated from the following file:

- gui/include/webBrowserTabUi.h

9.1073 wfs_photometry::WFSCentroid Class Reference

Classes

- struct object_t
- struct refList_t
- struct vector_t

Public Member Functions

- **WFSCentroid** (const ConfigureParams &configParams)

Protected Member Functions

- bool **readRefListFile** (bool bCheckApertures)
- void **centroid** (int n, int m, int niter, double thr, DoubleMatrix_t *pPsf, double *pX, double *pY)

Protected Attributes

- int **m_nIter**
- double **m_ron**
- double **m_background**
- long **m_thr**
- int **m_imageWidth**
- int **m_imageHeight**
- bool **m_bReadRefListFile**
- string **m_pRefListFile**
- int **m_refListCount**
- refList_t * **m_pRefList**
- int **m_allocatedObjects**
- int **m_objectCount**
- object_t * **m_pObject**
- vector_t * **m_pVector**

Friends

- class **WFSFit**

The documentation for this class was generated from the following file:

- het_lib/wfs_lib/include/wfs_photometry.h

9.1074 wfsData Class Reference

Public Types

- enum **format_t** {
  memory, **binary**, **text**, **fits**,
  fitsZ2O, **fitsO2Z**, **fitsLogical**, **fitsIndex**,
  ascii, **ds9** }

Public Member Functions

- virtual void **importData** (const string &, format_t formatType, uchar *=NULL, wfsData *=NULL)
- virtual int **exportData** (const string &, format_t formatType, uchar **=NULL, wfsData *=NULL)
- virtual void **newData** (int)
- virtual void **newData** (int, int)
- virtual void **newData** (int, int, int)
- virtual void **deleteData** (void)

Public Attributes

- bool **complete**

The documentation for this class was generated from the following file:

- het_lib/wfs_lib/include/wfslib.h

9.1075 WFSException Class Reference

Public Member Functions

- **WFSException** (tcsexcept::tcs_exception_t __wfs, const std::string &w, const char ∗file, int line, const char ∗func)

The documentation for this class was generated from the following file:

- het_lib/common/include/tcs_exception.h

9.1076 wfs_photometry::WFSFit Class Reference

Public Member Functions

- **WFSFit** (WFSCentroid *pCentroid, const ConfigureParams &configParams)
- void **doFit** (void)
- void **getOptions** (ConfigureParams &configParams)
- void **resetOptions** (const ConfigureParams &configParams)
- double **getXdecenter** (void)
- double **getYdecenter** (void)
- double **getFocus** (void)
- double **getTipAngle** (void)
- double **getTiltAngle** (void)
- double **getRhoAngle** (void)

Protected Member Functions

- void **readRecMatrixFile** (void)
- double **zernike37** (int index, double rho, double ang, double rho0)
- bool **invert** (DoubleMatrix_t *pMatrix1, unsigned int nRows, unsigned int nCols, DoubleMatrix_t *pMatrix2)
- bool **singularValueDecomposition** (DoubleMatrix_t *pA, unsigned int nRows, unsigned int nCols, DoubleVector_t *pW, DoubleMatrix_t *pV)
- bool **choleskyDecomposition** (DoubleMatrix_t *pA, unsigned int n, DoubleVector_t *pP)
- void **invertCholesky** (DoubleMatrix_t *pA, DoubleVector_t *pP, unsigned int n)

Protected Attributes

- WFSCentroid * **m_pCentroid**
- IntegerVector_t **m_subAppMask**
- DoubleMatrix_t **m_slopeFitMatrix**
- DoubleMatrix_t **m_invSlopeFitMatrix**
- ConfigureParams **m_configParams**
- bool **m_bRecMatrixFileRead**
- double **m_Xdecenter**
- double **m_Ydecenter**
- double **m_focus**
- double **m_tipAngle**
- double **m_tiltAngle**
- double **m_rhoAngle**

The documentation for this class was generated from the following file:

- het_lib/wfs_lib/include/wfs_photometry.h

9.1077 wfs_photometry::WFSFitException Class Reference

Public Member Functions

- **WFSFitException** (const string msg="WFSFit exception")
- virtual const char ∗ **what** ()

The documentation for this class was generated from the following file:

- het_lib/wfs_lib/include/wfs_photometry.h

9.1078 WFSHandler Class Reference

Public Member Functions

- **WFSHandler** (WFSLoop *l, const MessageHandlerDescriptor &m)

The documentation for this class was generated from the following file:

- tcs/metrology.h

9.1079 wfslib Class Reference

Public Member Functions

- void **shcentroid** ()
- void **gmoment** (int m, int n, float thresh, float **moment)
- void **ConvertImage** ()
- void **GetPixelValue** (uint offset, uint *value)
- void **statsnr** (uint nbins, float cutoff, bool **newmask, uint **ncount)
- void **median_filter** (int size)
- bool **pnpoly** (int nvert, float *vertx, float *verty, float testx, float testy)
- void **update_mask** ()
- void **update_imgmask** ()
- void **update_pplmask** ()
- void **build_imgmask** ()
- void **build_imgmask_thread** ()
- void **build_pplmask** ()
- void **build_pplmask_thread** ()
- void **finalize_grid** ()
- void **update_pupildata** (PupilData *dat)
- void **update_griddata** (SHGrid *grid)
- void **update_fitmatdata** (FitData &fitdat)
- void **build_modalmatrix** (ushort maxterm, ushort width, ushort height)
- void **build_modalmatrix** (ushort maxterm, ushort width, ushort height, double angle)
- void **build_slopematrix** ()
- void **build_fitmatrix** (double pix2mm, double pix2rad, double wavelen)
- void **buildnoisemap** (double **bgr, double **ron)
- void **measureslope_with_noisemap** (double **bgr, double **ron)
- void **measurefwhm_with_noisemap** (double **bgr, double **ron)
- void **fitpsf_with_noisemap** (double **bgr, double **ron, PSFTYPE ptype)
- void **rearrange_fitmatdata** (uint nvalid, bool *mask)
- void **estimate_fiterror** (int nvalid, bool **newmask, double **ron, gsl_vector *Error)
- void **calibrateslope** ()
- void **calibrateslope** (int nvalid, bool *newmask)
- void **solveformode** ()
- void **solveformode_orth** ()
- void **recon_fitmat** (gsl_matrix **A)
- void **build_orthonorm_fitmat** ()
- void **measurephotometrydata** ()
- void **orthonormalize** ()

Public Attributes

- NoiseAreaData **nadat**
- ImageData **imdat**
- TrackerData **trkdat**
- CentroidInfo **ceninfo**
- ImageMask **imgmask**
- ImageMask **obcmask**
- SHGrid **shgrid**

- PupilData **pupil**
- FitData **fitmat**
- FitData **ofitmat**
- SHTrackerData **shtrkdat**
- SHTrackerData **calibdat**
- ObscGeom **hetobsc**
- ObscGeom **obsc**
- PSFFitData **psffit**
- ModalMatrix **modalmat**
- OModalMatrix **omodalmat**
- SlopeMatrix **slopemat**
- ModalFitData **modalcof**
- PhotometryData **photdat**
- pthread_t * **thread**
- ulong **thread_start**
- ulong **thread_stride**
- BuildPupilThreadArgs * **thrarg**
- gsl_matrix * **psfparm**

Protected Member Functions

- void **pplmask_thread_worker** (ulong thrstart, ulong thrstride)
- void **imgmask_thread_worker** (ulong thrstart, ulong thrstride)

Static Protected Member Functions

- static void * **pplmask_thread_worker_callback** (void *arg)
- static void * **imgmask_thread_worker_callback** (void *arg)

The documentation for this class was generated from the following files:

- het_lib/wfs_lib/include/wfslib.h
- het_lib/wfs_lib/src/wfslib.cpp

9.1080 wfslib_het Class Reference

Public Member Functions

- **wfslib_het** (StaticConfig &pStaticConfig)
- void **Initialize** ()
- void **UpdateDynamicData** (bool pointingDataValid, itf_coordinate itf, bool instrumentDataValid, double current_arm_angle, double current_probe_angle, double current_x, double current_y, bool dmiDataValid, double dmi_distance, double imageUnixTime)
- void **Process** (const float *imageData, uint widthInPixels, uint heightInPixels)
- void **CommandWFS** (Command_t cmd)
- void **loadConfigfile** (string configfile, string tcscfgpath)
- void **loadFitsImage** (string fitsimage)
- void **loadFitsHeader** (string fitsimage)
- **wfslib_het** (StaticConfig &pStaticConfig)
- void **Initialize** ()
- void **UpdateDynamicData** (bool pointingDataValid, itf_coordinate itf, bool instrumentDataValid, double current_arm_angle, double current_probe_angle, double current_x, double current_y, bool dmiDataValid, double dmi_distance, double imageUnixTime)
- void **Process** (const float *imageData, uint widthInPixels, uint heightInPixels)
- void **CommandWFS** (Command_t cmd)

Public Attributes

- SHGrid **m1array**
- float **Xoff_asec**
- float **Yoff_asec**
- float **Woff_mm**
- float **Toff_asec**
- float **Poff_asec**
- float **GrocOff_um**
- float **seeing_mean**
- float **seeing_dev**

Protected Member Functions

- void **SetStaticData** (StaticConfig *cfg)
- void **SetDynamicData** (DynamicConfig *cfg)
- void **LoadStaticData** ()
- void **AlignMode** ()
- void **DeriveOffsets** ()
- void **PrintOffsets** ()
- void **RecordOffsets** ()
- void **InvalidateFitresults** ()
- void **MoveM1Geom** ()
- void **MoveTRKGeom** ()
- void **MoveSPDRGeom** ()
- void **MoveM4Geom** ()
- void **AffinegGeom** (ObscGeom *obj, float dx, float dy, float scale, float dclock)

- void **AffinegGeom** (SHGrid ∗obj, float dx, float dy, float scale, float dclock)
- void **BuildHETPupilMask** ()
- void **HETM1Mask** ()
- double **ComputeIllumination** ()
- void **FindOpticalAxisonwhichM1** ()
- void **SetStaticData** (StaticConfig ∗cfg)
- void **SetDynamicData** (DynamicConfig ∗cfg)
- void **LoadStaticData** ()
- void **AlignMode** ()
- void **DeriveOffsets** ()
- void **InvalidateFitresults** ()
- void **MoveM1Geom** ()
- void **MoveTRKGeom** ()
- void **MoveSPDRGeom** ()
- void **MoveM4Geom** ()
- void **AffinegGeom** (ObscGeom ∗obj, float dx, float dy, float scale, float dclock)
- void **AffinegGeom** (SHGrid ∗obj, float dx, float dy, float scale, float dclock)
- void **BuildHETPupilMask** ()
- void **HETM1Mask** ()
- double **ComputeIllumination** ()
- void **FindOpticalAxisonwhichM1** ()

Protected Attributes

- StaticConfig **static_cfg**
- DynamicConfig **dynamic_cfg**
- vector< std::string > **fitslist**
- ObscGeom **trker**
- ObscGeom **trker_original**
- ObscGeom **wfcm4**
- ObscGeom **wfcm4_original**
- ObscGeom **spidr**
- ObscGeom **spidr_original**
- ObscGeom **wfsobsc**
- SHGrid **m1array_original**
- SHGrid **wfsgrid_original**
- SHTrackerData **m1array_shtrkdat**
- uint **nbins**
- uint **nvalid**
- float **cutoff**
- bool ∗ **newmask**
- uint ∗ **ncount**
- double ∗ **bgr**
- double ∗ **ron**
- gsl_vector ∗ **variance**
- double **dx_m1**
- double **dy_m1**
- double **dclock_m1**
- double **dr_m4**
- double **dclock_m4**

- double **dx_trk**
- double **dy_trk**
- double **dclock_trk**
- double **dx_spd**
- double **dy_spd**
- double **dclock_spd**
- double **wfs_az**
- double **ppillum0**
- double **ppillum**
- double **DMIr**
- double **DMIa**
- uint **OAsegID**
- uint **DMIsegID**
- double **OAsegX**
- double **OAsegY**
- double **DMIsegX**
- double **DMIsegY**
- map< string, string > **cfgpar**
- map< string, string > **fitsHeader**
- itf_coordinate **itf**
- double **xoff**
- double **yoff**
- double **woff**
- double **toff**
- double **poff**
- double **goff**

The documentation for this class was generated from the following files:

- het_lib/wfs_lib/include/reducewfs.h
- het_lib/wfs_lib/include/wfslib_het.h
- het_lib/wfs_lib/src/wfslib_het.cpp

9.1081 wfsLogger Class Reference

Public Types

- enum **wfsLogLevel** {

  WFS_ERROR_NONE, **WFS_ERROR_DEBUG**, **WFS_ERROR_INFO**, **WFS_ERROR_-WARN**,

  WFS_ERROR_ERROR, **WFS_ERROR_FATAL**, **WFS_ERROR_ALARM** }

Static Public Member Functions

- static void **logError** (void ∗src, wfsLogLevel logLevel, const char ∗msg, const char ∗fileN, int lineN, const char ∗func)
- static void **issueException** (void ∗src, const char ∗msg, const char ∗fileN, int lineN, const char ∗func)

The documentation for this class was generated from the following files:

- het_lib/wfs_lib/include/wfs_error.h
- het_lib/wfs_lib/src/wfslib_het.cpp

9.1082 WFSLoop Class Reference

Public Member Functions

- **WFSLoop** (Instrument &instrument, const TCSServerConfig &config, TCSDriver *driver, Metrology *master=NULL)
- void **activate** (void)
- void **start** (void)
- void **start** (Message &m)
- void **set_focus** (double w)
- virtual void **apply_metrology** (Payload &p, const fptime &data_time, bool do_throw=0)
- void **set_tiptilt_window_length** (double length)
- unsigned **get_tiptilt_window_length** (void)
- void **set_focus_window_length** (double length)
- unsigned **get_focus_window_length** (void)
- void **set_tiptilt_gain** (double gain)
- double **get_tiptilt_gain** (void)
- void **set_focus_gain** (double gain)
- double **get_focus_gain** (void)
- void **center** (void)

The documentation for this class was generated from the following files:

- tcs/metrology.h
- tcs/metrology.cpp

9.1083 WFSToTCSLogLevelMap_t Struct Reference

Public Attributes

- wfsLogger::wfsLogLevel **WFSLevel**
- Logger::LogLevel **TCSLevel**

The documentation for this struct was generated from the following file:

- het_lib/wfs_lib/src/wfslib_het.cpp

9.1084 WriteOneWire Class Reference

Public Member Functions

- **WriteOneWire** (const string &appName, WriteOneWireConfig *config)
- void **Read** ()
- bool **FindWireSet** ()
- int **Write** (int cntl_id=-1, int device_idx=-1, int field_idx=-1, string data="")
- void **WriteAll** ()
- int **start** ()
- int **FindDevIdx** ()

The documentation for this class was generated from the following file:

- camra/write_one_wire/write_one_wire.cpp

9.1085 WriteOneWireConfig Class Reference

Public Member Functions

- **WriteOneWireConfig** (bool named=false)
- void print_summary (void)

  Called prior to displaying options.

- void set_options (OptionParser &o)

  Adds options to default set.

Public Attributes

- string **dsp_timing_file**
- int **cntl_id**
- string **ccd_id_left**
- string **ccd_id_right**
- string **ifu_id**
- string **spectrograph_slot**
- string **ifu_slot**

9.1085.1 Member Function Documentation

9.1085.1.1 void WriteOneWireConfig::print_summary (void) [inline, virtual]

This method can be used to describe the application. The banner printed is displayed above the command line options in the `-h` output.

Implements TCSConfig.

9.1085.1.2 void WriteOneWireConfig::set_options (OptionParser & *optionParser*) [inline, virtual]

Method that will be called to add options to the OptionParser prior to the parsing command line and/or configuration file(s).

Implements TCSConfig.

The documentation for this class was generated from the following file:

- camra/write_one_wire/write_one_wire.cpp

9.1086 WriteRegister Class Reference

9.1086.1 Detailed Description

Write to a modbus register for the pfip enclosure PLC.

Required parameters:

regnum<int> - The modbus register number to write. These numbers can be found in the PFIP enclosure hardware documentation.

regnum<int> - The modbus register number to write. These numbers can be found in the PFIP enclosure hardware documentation.

value<int> - An integer value to store in the register.

`#include <pfip_handler_doc.h>`

Public Member Functions

- **WriteRegister** (Receiver ∗rcv, ACQ_SubSystem ∗acq)
- virtual void HandlerAction (Message ∗pMessage)

  Do the work requested by the handler Each handler must implement.

- virtual void cleanupWork (Message ∗pMessage)

  Cleanup after performing handler action.

The documentation for this class was generated from the following files:

- pfip/include/pfipServer.h
- pfip/src/PFIPHandlers.cpp

9.1087 xsky_find Class Reference

9.1087.1 Detailed Description

The methods below assume radians for both input and output. The forward transform takes telecentric angles as measured on sky and returns telecentric angles as measured on the focal surface.

Public Types

- typedef double **value_type**

Public Member Functions

- double **operator()** (const FVector< double, 4 > &x, const FVector< double, 1 > &p, FVector< double, 1 > &jac) const

The documentation for this class was generated from the following file:

- het_lib/coord_trans/src/sky_fs.cpp

Chapter 10

File Documentation

10.1 het_lib/tthread/include/fast_mutex.h File Reference

10.1.1 Detailed Description

```
#include <pthread.h>
```

Include dependency graph for fast_mutex.h:This graph shows which files directly or indirectly include this file:

Classes

- class tthread::fast_mutex

Namespaces

- namespace tthread

10.2 het_lib/tthread/include/tinythread.h File Reference

10.2.1 Detailed Description

```
#include <errno.h>
#include <pthread.h>
#include <signal.h>
#include <sched.h>
#include <unistd.h>
#include <sys/time.h>
#include <ostream>
#include "tcs_clock.h"
```

Include dependency graph for tinythread.h:This graph shows which files directly or indirectly include this file:

Classes

- class tthread::mutex
- class tthread::recursive_mutex
- class tthread::lock_guard< T >
- class tthread::condition_variable
- struct tthread::_thread_start_info
- struct tthread::_thread_start_functor< T >
- struct tthread::_thread_start_functor_arg< F, A >
- struct tthread::_thread_start_member< 14463331 >
- struct tthread::_thread_start_member_arg< 14463331, A >
- class tthread::thread

  Thread class.

- class tthread::thread::id
- class tthread::ratio< N, D >
- class tthread::chrono::duration< _Rep, _Period >

Namespaces

- namespace tthread
- namespace tthread::chrono
- namespace tthread::this_thread

Defines

- #define **_TTHREAD_POSIX_**
- #define TINYTHREAD_VERSION_MAJOR 1

  TinyThread++ version (major number).

- #define TINYTHREAD_VERSION_MINOR 1

TinyThread++ version (minor number).

- #define TINYTHREAD_VERSION (TINYTHREAD_VERSION_MAJOR * 100 + TINYTHREAD_VERSION_MINOR)

  TinyThread++ version (full version).

- #define **_TTHREAD_DISABLE_ASSIGNMENT**(name)
- #define thread_local

Typedefs

- typedef void *(* tthread::pthread_trampoline)(void *)
- typedef long long **tthread::__intmax_t**
- typedef duration< __intmax_t, ratio< 1, 1000000000 > > tthread::chrono::nanoseconds

  Duration with the unit nanoseconds.

- typedef duration< __intmax_t, ratio< 1, 1000000 > > tthread::chrono::microseconds

  Duration with the unit microseconds.

- typedef duration< __intmax_t, ratio< 1, 1000 > > tthread::chrono::milliseconds

  Duration with the unit milliseconds.

- typedef duration< __intmax_t > tthread::chrono::seconds

  Duration with the unit seconds.

- typedef duration< __intmax_t, ratio< 60 > > tthread::chrono::minutes

  Duration with the unit minutes.

- typedef duration< __intmax_t, ratio< 3600 > > tthread::chrono::hours

  Duration with the unit hours.

Functions

- thread::id tthread::this_thread::get_id ()

  Return the thread ID of the calling thread.

- void tthread::this_thread::yield ()
- template<class _Rep , class _Period >
  void tthread::this_thread::sleep_for (const chrono::duration< _Rep, _Period > &aTime)

10.2.2 Define Documentation

10.2.2.1 #define _TTHREAD_DISABLE_ASSIGNMENT(name)

Value:

```
name(const name&); \
      name& operator=(const name&);
```

10.2.2.2 #define thread_local

Thread local storage keyword. A variable that is declared with the `thread_local` keyword makes the value of the variable local to each thread (known as thread-local storage, or TLS). Example usage:

```
// This variable is local to each thread.
thread_local int variable;
```

Note:

The `thread_local` keyword is a macro that maps to the corresponding compiler directive (e.g. `__declspec(thread)`). While the C++11 standard allows for non-trivial types (e.g. classes with constructors and destructors) to be declared with the `thread_local` keyword, most pre-C++11 compilers only allow for trivial types (e.g. `int`). So, to guarantee portable code, only use trivial types for thread local storage.
This directive is currently not supported on Mac OS X (it will give a compiler error), since compile-time TLS is not supported in the Mac OS X executable format. Also, some older versions of MinGW (before GCC 4.x) do not support this directive.

Index

~thread
    tthread::thread, 1260
_TTHREAD_DISABLE_ASSIGNMENT
    tinythread.h, 1477
__getattr__
    hetdex-dither::GuiderRelay, 597
__init__
    pytcs::tcs_event::TCSEvent, 1223
    pytcs::TCSLog::TCSLog, 1241
    tcs_event::TCSEvent, 1222
    TCSLog::TCSLog, 1240
    traj_class::Trajectory, 1307
_modbus, 193
_modbus_backend, 194
_modbus_rtu, 195
_modbus_tcp, 196
_modbus_tcp_pi, 197
_sft, 198
_threadpool_start_thread
    tthread, 189

Abort, 204
abort, 206
abortMove, 207
aboutDialog, 210
ACQ_Camera, 211
    Enable, 212
    GetACQCameraFilterPos, 212
    GetACQCameraStatus, 212
    Initialize, 212
    SetACQCameraFilter, 212
    SetACQCameraStatus, 212
ACQ_Shutter, 213
    ACQ_Shutter, 215
    ACQ_Shutter, 215
    CloseShutter, 215
    EnableShutter, 215
    GetShutterStatus, 215
    HomeShutter, 215
    Initialize, 215
    OpenShutter, 215
    SetShutterStatus, 215
    WriteShutterOpenClose, 216
ACQ_SubSystem, 217
    EnableDisableProbes, 220
    GetACQCameraFocus, 220
    SetACQCameraFilter, 220
    SetACQCameraFocus, 221
    SetACQCameraFocus_mm, 221
    SetCWFSCameraDeployed, 221
    SetDither, 221
    SetENWAFilterPosition, 221
    TakeExposure, 221
ACQCamInfo, 222
AcqCamStatus, 223
AcqCamStatus::camera_status, 273
AcqCamStatus::filter_status, 510
acqCamStatusTabUi, 225
AcqCWFSStatus, 226
AcqCWFSStatus::cwfs_status, 353
AcqENWAStatus, 227
AcqENWAStatus::enwa_status, 424
AcqMiscStatus, 228
AcqMiscStatus::misc_status, 747
AcqShutterStatus, 229
AcqShutterStatus::shutter_status, 1101
Acquisition and Guide Probe Assembly, 132
Acquisition and Probe Assembly, 140
activate, 230
    process, 230
add_correction, 231
    process, 231
add_system
    EventHandler, 441
    EventSink, 449
addTCSDataToFITSFile
    pasImagePipelineSubSystem, 825
AdjustDither, 232
Alarm_Monitor, 233
alertsDialog, 237
alrtlvl
    mbintf_bitfield_t, 720
    mbintf_inpreg_analog_t, 722
altaz_coordinate, 238
Amoeba, 239
    amotry, 239
amotry
    Amoeba, 239
Amp, 240
Amplifier related constants, 78

AnalysisRegion, 242
Apc, 67
APC Message Handlers, 71
APC Utility Classes, 72
apcCmd, 243
apcCmdConfig, 244
    print_summary, 244
    set_options, 244
apcConfig, 245
    print_summary, 245
    set_options, 245
APCData, 246
APCData::status, 1140
APCException, 247
apcPduSubSystem, 248
apcServer, 249
apcTabUi, 252
apcTabUi::tabData, 1198
aperture_phot
    tcs_photometry::Photometry, 875
apply_offset, 255
Archon, 256
ArchonHandler, 257
Array sizing constants, 76
array_dbr_int_t, 258
array_dbr_long_t, 259
ArrayType, 260
as_utc
    fptime, 530
assemble
    tcs_photometry::Photometry, 875
Att
    hetlib::hetutils, 171
    hetutils, 178
averager, 263
AVT, 264
AVT Camera, 112

BeginRangeProbes, 265
BestAzimuth
    hetlib::hetutils, 171
    hetutils, 178
bestDistance, 266
bit
    mbintf_bitfield_t, 720
Bitfield, 268
bitfields
    mbintf_hldreg_bitfield_t, 721
    mbintf_inpreg_bitfield_t, 724
bitfields_ct
    mbintf_hldreg_bitfield_t, 721
    mbintf_inpreg_bitfield_t, 724
BroadcastStatusEvents
    FCU_SubSystem, 473
BuildPupilThreadArgs, 269
BusyHandler, 270
    process, 270

Camera, 111, 271
CameraAVT, 274
CameraConfig, 275
    set_options, 277
cameraControlTabUi, 279
CameraFLI, 281
CameraFLI::CameraData, 280
cameraImageMarker, 282
cameraImageScene, 283
cameraImageTabUi, 287
cameraImageTabUiThreadWorker, 289
CameraInfo, 290
cameraMetrologyControlTabUi, 292
cameraMetrologyDataTabUi, 294
CameraPGR, 295
CameraSim, 297
CAMRA Classes, 80
CAMRA Configuration Classes, 73
CAMRA Handlers, 74
CAMRA Hardware Classes, 75
camra_handlers::abort, 205
    process, 205
camra_handlers::CAMRAHandler, 303
camra_handlers::close_shutter, 325
    process, 325
camra_handlers::command_controller, 333
    process, 333
camra_handlers::disable_pressure_gauges, 373
    process, 373
camra_handlers::disable_shutter, 376
    process, 376
camra_handlers::enable_pressure_gauges, 415
    process, 415
camra_handlers::enable_shutter, 417
    process, 417
camra_handlers::expose, 455
    process, 455
camra_handlers::get_ccd_temp, 535
    process, 535
camra_handlers::get_clock_voltages, 537
    process, 537
camra_handlers::get_cryo_pressure, 540
    process, 540
camra_handlers::get_cryo_temp, 542
    process, 542
camra_handlers::get_exposure_info, 547
    process, 547
camra_handlers::get_hardware_status, 554
    process, 554
camra_handlers::get_heater_voltage, 556

process, 556
camra_handlers::open_shutter, 807
process, 807
camra_handlers::override_ifu_associations, 810
process, 810
camra_handlers::populate_onewire_device_strings, 918
process, 918
camra_handlers::read_now, 989
process, 989
camra_handlers::reset, 999
process, 999
camra_handlers::reset_clock_voltages, 1001
process, 1001
camra_handlers::reset_mux, 1005
process, 1005
camra_handlers::reset_tcs, 1009
process, 1009
camra_handlers::set_bias_offset, 1030
process, 1030
camra_handlers::set_ccd_temp, 1032
process, 1032
camra_handlers::set_ccd_temp_xfer, 1034
process, 1034
camra_handlers::set_clock_voltages, 1036
process, 1036
camra_handlers::set_cryo_temp_xfer, 1039
process, 1039
camra_handlers::turn_off_leds, 1317
process, 1317
camra_handlers::wait_for_readout, 1436
process, 1436
camra_handlers::wait_for_shutter_close, 1439
process, 1439
camra_handlers::wait_for_write, 1441
process, 1441
CAMRAConfig, 298
print_summary, 298
set_options, 298
CAMRADriver, 300
exposure, 301
populate_onewire_device_strings, 301
reset_tcs, 301
stash_temps, 301
CAMRAException, 302
CAMRAHardware, 304
cancel_trajectory, 307
process, 307
TCSDriver, 1219
CBUF, 308
CCD, 310
CCD Array Management and Recording Application (CAMRA), 79
center, 312
process, 312
CentroidInfo, 313
check, 314
process, 314
check_variance
tcs_photometry::Photometry, 875
CheckNewCurrentEncoderPositions
Probe, 940
chunkHeader, 315
cJSON, 316
cJSON_Hooks, 317
Class Documentation, 104, 147, 150
CleanerConfig, 318
set_options, 318
clear_fiducial, 319
process, 319
clear_trajectories, 320
process, 320
TCSDriver, 1219
clearError, 321
Client, 323
clipToCircle_deg
hetlib::hetutils, 171
hetutils, 178
clipToCircle_rad
hetlib::hetutils, 171
hetutils, 178
Close, 324
close_shutter, 326
CloseShutter, 327
ACQ_Shutter, 215
cmd_code
sockcmd_t, 1112
cmdConfig, 328
print_summary, 328
set_options, 328
Cntl, 329
coefs_t, 331
Command, 332
command_controller, 334
Common Client-Side Logging, 92
Common Library, 85
ComputeMove
Probe, 940
Config, 337
config_root
TCSConfig, 1211
ConfigHandler, 338
Configuration, 65, 96, 118, 144, 148
configure
HETEarthOrientation, 608
TCSConfig, 1210
configureSystemsDialog, 342
configureTabsDialog, 343

confirmOffsetDialog, 345
Connect
    EPICSComm, 430
    MODBUSComm, 752
connect
    pytcs::tcs_event::TCSEvent, 1223
    tcs_event::TCSEvent, 1222
    TCSSocket, 1246
connected
    TCSSocket, 1246
consoleConfig, 347
    print_summary, 347
    set_options, 347
Controller initialization constants, 77
ControllerConfig, 348
    print_summary, 348
    set_options, 348
Coordinate Systems and Transformations, 87
Coordinate Transform Library, 90
Coordinate Transforms and Structure Mount Model, 89
CoordinateTransformException, 349
copy_square
    tcs_photometry::Photometry, 875
correct_for_structure, 350
    process, 350
Correction, 351
cubFit, 352
CWFSInfo, 354
CyclePipeline, 355

data
    Payload, 836
dataChunk, 358
deactivate, 359
    process, 359
debug
    TCSConfig, 1211
deep_arg_t, 360
degrees_to_hours
    hetlib::hetutils, 171
    hetutils, 178
delta_time, 362
DeployACQCameraMirror, 363
DeployCWFSCamera, 364
DeployCWFSCameraMirror, 365
DeployPVCamera, 366
desc
    mbintf_bitfield_t, 720
    mbintf_inpreg_analog_t, 722
description
    moveRTF, 769
    moveTSF, 774
detach
    tthread::thread, 1260
detachedTabWindow, 367
DeviceConfig, 368
    set_options, 368
DevicePipelineEnableCCASDMIHandlerClass, 369
    process, 369
DevicePipelineEnableSACDMIHandlerClass, 370
    process, 370
DevicePipelineSetGROCHandlerClass, 371
    process, 371
DevicePipelineStatusHandlerClass, 372
    process, 372
disable
    EventSink, 449
disable_pressure_gauges, 374
disable_rho, 375
    process, 375
disable_shutter, 377
DisableProbes, 378
disconnect
    TCSSocket, 1246
DMI_get_status, 379
    process, 379
DMIData, 381
DMIData::status, 1155
DMIHandler, 382
DMILoop, 383
dmiTabUi, 386
Doh, 387
Dome, 97, 98, 101, 102
dome_Abort, 388
dome_GetPosition, 389
dome_Home, 390
dome_IssuePMACCmd, 391
dome_MoveTo, 392
dome_MoveToCCAS, 393
dome_OnLine, 394
dome_Reset, 395
dome_SetDirection, 396
dome_SetPower, 397
dome_SetVelocity, 398
dome_Status, 399
dome_Stop, 400
dome_Stow, 401
dome_Test, 402
DomeClient, 403
DomeConfig, 404
    set_options, 404
DomeStatus, 405
DomeStatus::status, 1134
DomeStatusHandlerClass, 406
    process, 406
domeTabUi, 408

done
Message, 732
drawHexagon3D
GLTrackerDriver, 584
drawPrimaryMirror
GLTrackerDriver, 584
droppablePlainTextEdit, 409
DynamicConfig, 411

editPreferencesDialog, 412
elliptical_annulus_mask
tcs_photometry::Photometry, 875
elliptical_mask
tcs_photometry::Photometry, 875
Enable
ACQ_Camera, 212
enable_pressure_gauges, 414
enable_rho, 416
process, 416
enable_shutter, 418
EnableCCASDMI, 419
EnableDisableProbes
ACQ_SubSystem, 220
ProbeAssembly, 942
EnableLamps
FCU_SubSystem, 473
EnableProbes, 420
EnableSACDMI, 421
EnableShutter
ACQ_Shutter, 215
EndRangeProbes, 422
Enum, 423
ENWAInfo, 425
Ephemeris, 426
EphList, 427
EphRec, 428
EPICSComm, 429
Connect, 430
EPICSComm, 430
GetString, 430
getValues, 430
SetInt, 430
setValues, 430
EPICSException, 432
EPICSHandle, 433
EPICSRelayConfig, 434
print_summary, 434
set_options, 434
EPICSRelayDriver, 435
EPICSVariables, 143
eq_coords, 436
equ2hor
equ_hor, 169
equ_hor, 169
equ2hor, 169
Event, 437
operator<<, 437
variable, 437
eventCmd, 440
EventHandler, 441
add_system, 441
set_default, 441
set_global_defaults, 441
value, 442
wait, 442
EventHeader, 443
EventList, 444
eventMonitorTabUi, 446
eventMonitorTabUiThreadWorker, 447
eventsConfig, 448
print_summary, 448
set_options, 448
EventSink, 449
add_system, 449
disable, 449
EventSink, 449
next_event, 449
set_filter, 449
EventSource, 451
name, 451
EventSourceRoot, 452
EventSourceSocket, 453
ExportData, 454
expose, 456
Exposure, 457
exposure
CAMRADriver, 301
ExposureDescriptor, 458
ExposureInfo, 460
ExposureInfo::BINNING, 267
ExposureInfo::FILTER_INFO, 509
ExposureInfo::Image, 633
ExposureInfo::PlateScale, 899
ExposureInfo::ROI, 1014
ExposureInfo::Sensor, 1024
ExposureSetup
FCU_SubSystem, 473

Facility Calibration Unit, 141
fast_aperture_phot
tcs_photometry::Photometry, 875
fast_fixed_aperture_phot
tcs_photometry::Photometry, 876
fast_kron_aperture_phot
tcs_photometry::Photometry, 876
FCU_Lamp, 464
FCU_LDLS, 465
FCU_Shutter, 466

FCU_Slider, 467
FCU_State, 468
FCU_SubSystem, 472
    BroadcastStatusEvents, 473
    EnableLamps, 473
    ExposureSetup, 473
    MoveOutputLightGuide, 473
    MoveSelectorPosition, 473
    ReplyCurrentExposure, 473
    ReplyLastExposure, 473
    SetValue, 474
FCUCancelExposure, 475
FCUClasses, 133
FCUCommException, 476
FCUCommunication, 477
fcuControlTabUi, 479
FCUDeployHead, 480
FCUDisableAllLights, 481
FCUEnableLamps, 482
FCUEnableLightSources, 483
FCUException, 484
FCUExposeLDLS, 485
FCUExposeSlider
    FCUHandler, 490
FCUExposeSliderA, 486
FCUExposeSliderB, 487
FCUGetCurrentExposureInfo, 488
FCUGetLastExposureInfo, 489
FCUHandler, 490
    FCUExposeSlider, 490
    FCUHandler, 490
    parseSlider, 490
    parseSliderPos, 491
FCUInitLightSources, 492
FCUMoveLightBank, 493
FCUMoveLightGuide, 494
FCUMoveSlider, 495
FCUMutexException, 496
FCUPrintRegister, 497
FCURetractHead, 498
FCUSetLampWarmUp, 499
FCUSetLDLSWarmUp, 500
FCUShutDown, 501
FCUStatus, 502
FCUStatus::fcu_status, 470
fcuStatusTabUi, 504
FCUWriteRegister, 505
fiducialDialog, 506
Filters, 511
find
    hetlib::spectrograph::SpectrographList, 1123
find_radec
    pt_itf, 182
find_stars
    tcs_photometry::Photometry, 876
fit_bivariate_gaussian
    tcs_photometry::Photometry, 876
fit_bivariate_moffat
    tcs_photometry::Photometry, 876
FitData, 512
FITS Image Storage, 127
FitsOutWrapper, 515
fixed_aperture_phot
    tcs_photometry::Photometry, 876
fixed_point::fixed_internal::ioflags, 667
fixed_point::fixed_internal::power10, 921
fixed_point::fixed_internal::power10< 0 >, 922
fixed_point::fixed_internal::power10_aux, 923
fixed_point::fixed_internal::power10_aux< N, true >, 924
fixed_point::fixed_internal::square, 1125
fixed_point::FixedPoint, 516
    FixedPoint, 517
FixedPoint
    fixed_point::FixedPoint, 517
flagValues, 519
FLI Camera, 113
flush
    TCSSocket, 1246
fmtConsole
    Logger, 705
focalPlaneShutterScene, 520
focalPlaneShutterTabUi, 522
focus_model_low
    MountModel, 761
Foo, 523
FooBar, 524
force_state_event, 525
    process, 525
ForceStatusEvent, 526
ForceStatusEventsHandlerClass, 527
    process, 527
formatChunk, 528
fptime, 529
    as_utc, 530
    fptime, 529, 530
    from_decimal_equinox, 530
    now, 530
from_decimal_equinox
    fptime, 530
fw102b::FW102B, 531

geHeaters::Ccd, 309
GenerateExceptionHandler, 532
    process, 532
get, 533
    process, 533
get_analysis_region, 534

process, 534
get_ccd_temp, 536
get_clock_voltages, 538
get_correction, 539
process, 539
get_cryo_pressure, 541
get_cryo_temp, 543
get_current_local_sidereal_time, 544
process, 544
get_current_pointing, 545
process, 545
get_current_trajectory_info, 546
process, 546
get_exposure_info, 548
get_fiducial, 549
process, 549
get_focus_gain, 550
process, 550
get_focus_offset, 551
process, 551
get_focus_window_length, 552
process, 552
get_hardware_status, 553
get_heater_voltage, 555
get_itf_offset, 557
process, 557
get_messages
IglooRegAlertList, 622
get_metadata, 558
process, 558
get_name
mbreg_t, 726
get_next_trajectory_info, 559
process, 559
get_origin
Message, 732
get_radec_offset, 560
process, 560
get_radec_offset_register
TCSDriver, 1219
get_receive_time
Message, 732
get_sky_offset, 561
process, 561
get_status, 562
process, 562
get_tiptilt_gain, 563
process, 563
get_tiptilt_offset, 564
process, 564
get_tiptilt_window_length, 565
process, 565
get_tuning, 566
process, 566
GetACQCameraFilterPos
ACQ_Camera, 212
GetACQCameraFocus
ACQ_SubSystem, 220
GetACQCameraStatus
ACQ_Camera, 212
GetACQSystemInfo, 567
GetAllProbeEncoderCounts
ProbeAssembly, 942
GetAnalysisRegion, 568
getCurrentIndexTime
st3, 184
GetEncoderCount
ProbePosition, 954
GetExposureTime, 569
GetGain, 570
GetLightSourcePowerLevel, 571
GetOverScan, 572
GetPhotPhotometryOptions, 573
GetPosition, 574
getposition
tcsutils, 186
GetProbeInfo, 575
ProbeAssembly, 942
GetProbePosition, 576
GetRegionOfInterest, 577
GetShutterStatus
ACQ_Shutter, 215
GetString
EPICSComm, 430
GetTemperatures, 578
getValues
EPICSComm, 430
MODBUSComm, 753
GetWeatherData, 579
GetWFSPhotometryOptions, 580
globals, 581
GLTrackerConfig, 582
GLTrackerDriver, 583
drawHexagon3D, 584
drawPrimaryMirror, 584
GLTrackerDriver::tracker_position, 1292
go_next, 585
process, 585
TCSDriver, 1219
GoNextFlags, 586
GroupConfig, 587
print_summary, 587
set_options, 587
guiConfig, 588
print_summary, 589
set_options, 589
guideBoxDialog, 591
GuideLoop, 593

Guider_photometry
tcs_photometry::Photometry, 876
GuiderHandler, 595
GUIException, 598
guiPreferences, 599
guiSound, 600

h
moveRTF, 770
moveTSF, 775
handler_t, 602
hardware_concurrency
tthread::thread, 1260
HardwareElement, 603
headerChunk, 604
HelloFunctor, 605
help
TCSConfig, 1211
HelpHandler, 606
process, 606
HET Earth orientation prediction, 82
het_lib/tthread/include/fast_mutex.h, 1475
het_lib/tthread/include/tinythread.h, 1476
hetdex-dither::GuiderRelay, 597
__getattr__, 597
hetdex-dither::HetdexDitherError, 607
HETEarthOrientation, 608
configure, 608
lst, 608
lst_str, 608
pmx, 608
HETEarthOrientationException, 609
HETillumination, 610
HETillumination::M1motion, 711
HETillumination::M4motion, 712
HETillumination::Spidermotion, 1124
HETillumination::Trackermotion, 1299
hetlib::hetutils, 170
Att, 171
BestAzimuth, 171
clipToCircle_deg, 171
clipToCircle_rad, 171
degrees_to_hours, 171
Ho, 171
hours_to_degrees, 172
hours_to_radians, 172
modulo, 172
Pa, 172
radians_to_hours, 172
Tde, 172
hetlib::Points, 173
knownLists, 173
hetlib::spectrograph::Spectrograph, 1118
init_FmKeyStr, 1118
mkDisplay, 1119
Pad, 1119
parse_KeyStr, 1119
PowerStr, 1119
hetlib::spectrograph::SpectrographList, 1123
find, 1123
sortSpect, 1123
hetlib::tcsconfig, 175
hetlib::tcsutils, 176
start_clients, 176
hetutils, 177
Att, 178
BestAzimuth, 178
clipToCircle_deg, 178
clipToCircle_rad, 178
degrees_to_hours, 178
Ho, 179
hours_to_degrees, 179
hours_to_radians, 179
modulo, 179
Pa, 179
radians_to_hours, 179
Tde, 179
hex_fit, 612
Ho
hetlib::hetutils, 171
hetutils, 179
Home, 613
HomeProbes, 614
HomeShutter, 615
ACQ_Shutter, 215
hostname
TCSConfig, 1211
HotPixel_t, 616
hours_to_degrees
hetlib::hetutils, 172
hetutils, 179
hours_to_radians
hetlib::hetutils, 172
hetutils, 179

ICSConfig, 617
name, 617
ICSDaemon, 618
ICSQuery, 619
id
Message, 732
sockcmd_reg_id_t, 1111
tthread::thread::id, 620
IglooData, 621
IglooData::status, 1158
IglooRegAlertList, 622
get_messages, 622
initialize, 622

IglooSensors, 623
IglooSensorsConfig, 624
    print_summary, 624
    set_options, 624
IglooSensorsConnection, 625
IglooSensorsException, 626
IglooSensorsHandler, 627
IglooStatusField, 628
IglooStatusFieldList, 629
iglooStatusFieldWidgetSet, 630
    iglooStatusFieldWidgetSet, 630
iglooTabUi, 632
Image, 634
Image Pdu, 69
Image pdu Sub-System, 68
Image Pipeline, 124
Image Pipeline Sub-System, 123
Image Processor, 116, 120
Image Storage, 122
ImageCleaner, 635
ImageData, 636
ImageMask, 638
ImagePipelineGetExposureTimeHandlerClass, 639
    process, 639
ImagePipelineGetGainHandlerClass, 640
    process, 640
ImagePipelineGetOverScanHandlerClass, 641
    process, 641
ImagePipelineGetPhotPhotometryOptionsHandlerClass, 642
    process, 642
ImagePipelineGetRegionOfInterestHandlerClass, 643
    process, 643
ImagePipelineGetWFSPhotometryOptionsHandlerClass, 644
    process, 644
ImagePipelineSetCleanImagesHandlerClass, 645
    process, 645
ImagePipelineSetCleaningBiasTypeHandlerClass, 646
    process, 646
ImagePipelineSetFilterExposureTimeFactorHandlerClass, 647
    process, 647
ImagePipelineSetFilterSequenceHandlerClass, 648
    process, 648
ImagePipelineSetFilterZeroPointHandlerClass, 649
    process, 649
ImagePipelineSetObjectAndMagnitudesHandlerClass, 650
    process, 650
ImagePipelineSetOverScanHandlerClass, 651
    process, 651
ImagePipelineSetPhotometryOptionsHandlerClass, 652
    process, 652
ImagePipelineSetPhotPhotometryOptionsHandlerClass, 653
    process, 653
ImagePipelineSetProcessingTypeHandlerClass, 654
    process, 654
ImagePipelineSetWFSPhotometryOptionsHandlerClass, 655
    process, 655
ImagePipelineStartSendingImagesHandlerClass, 656
    process, 656
ImagePipelineStatus, 657
ImageProcessor, 658
imageRect, 659
index_time, 660
inEllipse
    pt_itf, 182
init_FmKeyStr
    hetlib::spectrograph::Spectrograph, 1118
init_t, 661
Initialize
    ACQ_Camera, 212
    ACQ_Shutter, 215
    ProbeAssembly, 942
initialize
    IglooRegAlertList, 622
initTMCS, 662
inject_metrology, 664
    process, 664
Instrument, 665
    name, 665
integrate_soft_e_mask_fixed
    tcs_photometry::Photometry, 876
interp_ephem
    Point, 908
interrupt
    Message, 732
interrupt_mark
    Message, 732
InterruptHandler, 666
    process, 666
is_done
    Message, 732
is_error
    Message, 732
is_hld
    sockcmd_reg_id_t, 1111
is_running
    tthread::ThreadWorker, 1270
IsProbeEnabled
    PFIPProbeInfo, 867

IssuePMACCmd, 668
iterate
    Point, 908
itf_coordinate, 669
itf_to_rtf
    MountModel, 761
itf_tsf, 670

join
    tthread::thread, 1261
joinable
    tthread::thread, 1261
julian_day, 671

knownLists
    hetlib::Points, 173
    Points, 180
kron_aperture_phot
    tcs_photometry::Photometry, 877

label
    mbintf_bitfield_t, 720
    mbintf_hldreg_bitfield_t, 721
    mbintf_inpreg_analog_t, 722
    mbintf_inpreg_bitfield_t, 724
Legacy Servers Access System (Legacy), 103
legacy_ForceStatusEvents, 674
legacyConfig, 675
    print_summary, 675
    set_options, 675
legacyDomeSubSystem, 677
legacyDomeSubSystem::status, 1157
LegacyException, 680
legacyServer, 681
legacyShutterSubSystem, 683
legacyShutterSubSystem::status, 1152
legacyStructureSubSystem, 686
legacyStructureSubSystem::status, 1159
legacyTabUi, 691
legacyWeatherSubSystem, 692
legacyWeatherSubSystem::statistic, 1131
legacyWeatherSubSystem::status, 1138
liborchestrate::DmiSubsystem, 384
liborchestrate::GuiSubsystem, 601
liborchestrate::LegacyDomeSimulator, 676
liborchestrate::LegacyShutterSimulator, 682
liborchestrate::LegacyStructureSimulator, 685
liborchestrate::LegacySubsystem, 689
liborchestrate::MonitorSubsystem, 758
liborchestrate::Named, 781
liborchestrate::PasSubsystem, 833
liborchestrate::PfipSubsystem, 871
liborchestrate::PlotterTcsSubsystem, 903
liborchestrate::PlotterTrackerSubsystem, 904
liborchestrate::Subsystem, 1191
liborchestrate::SubsystemManager, 1195
liborchestrate::TcsSubsystem, 1251
liborchestrate::TmcsSubsystem, 1279
liborchestrate::TrackerSubsystem, 1304
Light Source, 117, 119
LightSource, 695
LightSourceConfig, 696
    set_options, 696
limit_lower
    mbintf_inpreg_analog_t, 722
limit_upper
    mbintf_inpreg_analog_t, 722
linFit, 697
load_measurement
    PIDLoop, 884
load_trajectory, 698
    process, 698
loadedUi, 699
loadMarkersDialog, 700
lock
    tthread::fast_mutex, 462
    tthread::mutex, 778
    tthread::recursive_mutex, 994
log_levels
    TCSConfig, 1211
LogClientException, 703
LogErrors
    PFIPProbeInfo, 867
Logger, 704
    fmtConsole, 705
    makeFileLoc, 705
    SendExternalLogMessage, 705
    SendLogMessage, 705
LoggerDB, 707
LogRelayConfig, 708
LogRelayDriver, 709
    process, 709
LogServerException, 710
lst
    HETEarthOrientation, 608
lst_str
    HETEarthOrientation, 608

m7302::M7302, 713
m_bit_id
    mbreg_t, 726
m_is_hld
    mbreg_t, 726
m_regptr
    mbreg_t, 726
mainWindow, 714
mainWindow::DashBoardAlerts, 356
makeFileLoc

Logger, 705
mark_pixels
tcs_photometry::Photometry, 877
MAXONStatusInfo, 719
mbintf_bitfield_t, 720
alrtlvl, 720
bit, 720
desc, 720
label, 720
normal, 720
mbintf_hldreg_bitfield_t, 721
bitfields, 721
bitfields_ct, 721
label, 721
mreg_id, 721
reg_id, 721
mbintf_inpreg_analog_t, 722
alrtlvl, 722
desc, 722
label, 722
limit_lower, 722
limit_upper, 722
reg_id, 722
regval_div, 722
regval_mult, 722
mbintf_inpreg_bitfield_t, 724
bitfields, 724
bitfields_ct, 724
label, 724
reg_id, 724
mbreg_ptr_t, 725
mbreg_t, 726
get_name, 726
m_bit_id, 726
m_is_hld, 726
m_regptr, 726
mbregid_t, 728
MBRegVal, 729
md5::MD5, 730
Message, 731
done, 732
get_origin, 732
get_receive_time, 732
id, 732
interrupt, 732
interrupt_mark, 732
is_done, 732
is_error, 732
respond, 732
Message Handler Documentation, 146
MESSAGE_SETJMP
remote_signal, 93
MessageHandler, 734
process, 734
MessageHandlerDescriptor, 736
MessageHeader, 737
Messaging Library, 94
MessagingException, 738
metadata_entry, 739
MetaPayload, 740
Metrology, 741
MetrologyHandler, 742
MetrologyLoop, 743
MetrologyPIDLoop, 744
MidRangeProbes, 745
MirrorDistance, 746
MiscInfo, 748
mkDisplay
hetlib::spectrograph::Spectrograph, 1119
ModalFitData, 749
ModalMatrix, 750
MODBUS Utility Data Structures, 134
modbus_mapping_t, 751
MODBUSComm, 752
Connect, 752
getValues, 753
setValues, 753
MODBUSTime, 754
modified_julian_day, 755
modulo
hetlib::hetutils, 172
hetutils, 179
MonitorConfig, 756
MonitorDriver, 757
process, 757
MonPrintRegister, 759
MonWriteRegister, 760
MountModel, 761
focus_model_low, 761
itf_to_rtf, 761
pointing_mount_model_offset, 761
rotateXY, 761
rtf_to_itf, 761
mountModel, 763
MountModelException, 764
MoveOutputLightGuide
FCU_SubSystem, 473
MoveProbeParms, 765
MoveProbes, 766
MoveProbes_Enc, 767
MoveProbes_RaDec, 768
moveRTF, 769
description, 769
h, 770
pos_dict, 770
MoveSelectorPosition
FCU_SubSystem, 473
moveStructure

tcsutils, 186
MoveTo, 772
MoveToStructAz, 773
moveTSF, 774
description, 774
h, 775
pos_dict, 775
mreg_id
mbintf_hldreg_bitfield_t, 721
MsgAttribute, 777
Mux, 780

name
EventSource, 451
ICSConfig, 617
Instrument, 665
TCSConfig, 1212
named_lookup
tcsutils, 186
namedClient, 782
namedConfig, 783
print_summary, 783
set_options, 783
NamedException, 784
namedHandler, 785
process, 785
namedServer, 787
native_handle
tthread::thread, 1261
native_wrapper, 788
next
pytcs::tcs_event::TCSEvent, 1223
tcs_event::TCSEvent, 1222
next_event
EventSink, 449
NoiseAreaData, 789
noncopyable, 790
NOOPHandler, 791
process, 791
normal
mbintf_bitfield_t, 720
notify_all
tthread::condition_variable, 335
notify_one
tthread::condition_variable, 335
now
fptime, 530

ObscGeom, 795
OCDStatus, 796
OCDStatus::status, 1147
offset_fiducial, 797
process, 797
offset_focus, 798
process, 798
offset_itf, 799
process, 799
offset_probe, 800
process, 800
offset_sky, 801
process, 801
offset_tiptilt, 802
process, 802
offset_trajectory, 803
process, 803
OModalMatrix, 804
OnLine, 805
Open, 806
open_shutter, 808
OpenShutter, 809
ACQ_Shutter, 215
operator<<
Event, 437
Payload, 836
optAz
Point, 908
Outlet, 66
override_ifu_associations, 811

P, 812
process, 812
Pa
hetlib::hetutils, 172
hetutils, 179
Pad
hetlib::spectrograph::Spectrograph, 1119
ParmsIterator, 813
ParmsObject, 814
parse_KeyStr
hetlib::spectrograph::Spectrograph, 1119
parseSlider
FCUHandler, 490
parseSliderPos
FCUHandler, 491
PAS Message Handlers, 129
PAS Utility Classes, 130
PASCameraClient, 815
pasConfig, 816
print_summary, 816
set_options, 816
pasDevicePipelineSubSystem, 817
pasDevicePipelineSubSystem::DMIData, 380
pasDevicePipelineSubSystem::SAMSData, 1020
PASException, 819
pasImageCleanerStandard, 820
pasImagePipelineSubSystem, 821
addTCSDataToFITSFile, 825
pasImagePipelineSubSystem::FITSMapEntry, 514

pasImagePipelineSubSystem::ImageHandlerData, 637
pasImagePipelineSubSystem::PFIPData, 857
pasImagePipelineSubSystem::TCSData, 1215
pasImagePipelineSubSystem::TCSData::fiducialObject, 508
pasImageProcessorPhot, 826
pasImageProcessorWFS, 827
pasImageStorage, 828
pasImageStorageFITS, 829
pasPipelineSubSystem, 830
pasServer, 832
pasTTLaserLightSource, 834
Payload, 835
    data, 836
    operator<<, 836
    set_data, 836
    set_parm, 837
    size, 837
Payload Alignment System (APC), 70
Payload Alignment System (PAS), 128
Payload::Data, 357
pduConfig, 838
    set_options, 838
PduOnLine, 839
PFIP Configuration Settings, 137
PFIP Interface Operations and Events, 139
pfip_clear_positions, 840
    process, 840
pfip_close_shutter, 841
    process, 841
pfip_coord, 842
    pfip_coord, 842
    pfip_coord, 842
    telecen2probe, 842
PFIP_EPICSVariable, 844
pfip_insert_acquisition_pickoff, 845
    process, 845
PFIP_Monitor, 846
    PFIP_Monitor, 846
    PFIP_Monitor, 846
pfip_move_probes, 847
    process, 847
pfip_open_shutter, 848
    process, 848
pfip_withdraw_acquisition_pickoff, 849
    process, 849
PFIPClient, 850
PFIPClientInstrumentHandler, 851
PFIPConfig, 852
    print_summary, 854
    set_options, 854
pfipControlTabUi, 856
pfipEnvironmentTabUi, 859
PFIPException, 860
PFIPHandler, 861
    PFIPHandler, 861
PFIPInterrupt, 862
PFIPLampsDB, 863
PFIPModbusException, 864
PFIPMonitorStatus, 865
PFIPMonitorStatus::alarms, 235
PFIPMonitorStatus::status, 1154
PFIPProbeInfo, 866
    IsProbeEnabled, 867
    LogErrors, 867
    WaitForProbeEnabled, 867
pfipServer, 868
    pfipServer, 868
PFIPServer Classes, 138
PFIPStatusActions, 869
    PFIPStatusActions, 869
PFIPStatusBase, 870
PFIPStatusBase::flags, 518
PFIPStatusBase::status, 1139
Photometry Image Processor, 125
PhotometryData, 879
photometryOptionsDialog, 881
PIDHandler, 882
PIDLoop, 883
    load_measurement, 884
    set_proportional, 884
pingHandler, 885
Pipe Line, 121
PipelineConfig, 886
    set_options, 887
PipelineCyclePipelineHandlerClass, 888
    process, 888
PipelineOnLine, 889
PipelineRelinquishControlHandlerClass, 890
    process, 890
PipelineSetCycleTimeHandlerClass, 891
    process, 891
PipelineSetIntervalSpeedHandlerClass, 892
    process, 892
PipelineStartPipelineHandlerClass, 893
    process, 893
PipelineStatus, 894
PipelineStatus::status, 1133
PipelineTakeControlHandlerClass, 895
    process, 895
Pixel, 897
Pixel_t, 898
plotEventsTabUi, 900
PlotterConfig, 901
PlotterDriver, 902
    process, 902
pmx

HETEarthOrientation, 608
Point, 905
interp_ephem, 908
iterate, 908
optAz, 908
Point, 907
printEL, 908
printInversePointing, 908
tracks_, 909
point, 910
Point_t, 911
PointGrey Camera, 114
Pointing Servers Access System (Pointing), 149
pointing_coordinate, 912
pointing_ForceStatusEvents, 913
pointing_mount_model_offset
MountModel, 761
pointingConfig, 914
print_summary, 914
set_options, 914
pointingServer, 915
Points, 180
knownLists, 180
Pool, 916
Pool, 917
populate_onewire_device_strings, 919
CAMRADriver, 301
pos_dict
moveRTF, 770
moveTSF, 775
pos_fit, 920
PowerACQCamera, 925
PowerBIBCamera, 926
PowerHPF_ACQCamera, 927
PowerHR_SASCamera, 928
PowerOffAll, 929
PowerOnAll, 930
PowerProbeCamera, 931
PowerPV_CWFSCamera, 932
PowerSBC1, 933
PowerSBC2, 934
PowerStr
hetlib::spectrograph::Spectrograph, 1119
PowerSwitch1, 935
PowerSwitch2, 936
PowerTTCamera, 937
Prime Focus Instrument Package Server (PFIPServer), 135
print_summary
apcCmdConfig, 244
apcConfig, 245
CAMRAConfig, 298
cmdConfig, 328
consoleConfig, 347
ControllerConfig, 348
EPICSRelayConfig, 434
eventsConfig, 448
GroupConfig, 587
guiConfig, 589
IglooSensorsConfig, 624
legacyConfig, 675
namedConfig, 783
pasConfig, 816
PFIPConfig, 854
pointingConfig, 914
ProxyConfig, 968
sampleConfig, 1017
SpectrographConfig, 1122
SubSystemConfig, 1192
TCSConfig, 1211
TCSServerConfig, 1244
TrackerConfig, 1294
WriteOneWireConfig, 1472
printEL
Point, 908
printInversePointing
Point, 908
Probe, 938
CheckNewCurrentEncoderPositions, 940
ComputeMove, 940
Probe, 939
ProbePosition::EncoderCountsToSphereRadians, 940
Probe Assembly classes, 142
ProbeAssembly, 941
EnableDisableProbes, 942
GetAllProbeEncoderCounts, 942
GetProbeInfo, 942
Initialize, 942
ReadStatusValues, 942
UpdateProbePositions, 942
probeassembly
WhichMotor, 142
probeAssemblyScene, 944
probeAssemblyTabUi, 946
probeControlTabUi, 948
probeData, 949
probeGainControlTabUi, 951
ProbeInfo, 952
ProbeItem, 953
ProbePosition, 954
GetEncoderCount, 954
SphereRadiansToEncoderCounts, 955
ProbePosition::EncoderCountsToSphereRadians
Probe, 940
ProbeStatus, 956
probeStatusTabUi, 958
process

activate, 230
add_correction, 231
BusyHandler, 270
camra_handlers::abort, 205
camra_handlers::close_shutter, 325
camra_handlers::command_controller, 333
camra_handlers::disable_pressure_gauges, 373
camra_handlers::disable_shutter, 376
camra_handlers::enable_pressure_gauges, 415
camra_handlers::enable_shutter, 417
camra_handlers::expose, 455
camra_handlers::get_ccd_temp, 535
camra_handlers::get_clock_voltages, 537
camra_handlers::get_cryo_pressure, 540
camra_handlers::get_cryo_temp, 542
camra_handlers::get_exposure_info, 547
camra_handlers::get_hardware_status, 554
camra_handlers::get_heater_voltage, 556
camra_handlers::open_shutter, 807
camra_handlers::override_ifu_associations, 810
camra_handlers::populate_onewire_device_strings, 918
camra_handlers::read_now, 989
camra_handlers::reset, 999
camra_handlers::reset_clock_voltages, 1001
camra_handlers::reset_mux, 1005
camra_handlers::reset_tcs, 1009
camra_handlers::set_bias_offset, 1030
camra_handlers::set_ccd_temp, 1032
camra_handlers::set_ccd_temp_xfer, 1034
camra_handlers::set_clock_voltages, 1036
camra_handlers::set_cryo_temp_xfer, 1039
camra_handlers::turn_off_leds, 1317
camra_handlers::wait_for_readout, 1436
camra_handlers::wait_for_shutter_close, 1439
camra_handlers::wait_for_write, 1441
cancel_trajectory, 307
center, 312
check, 314
clear_fiducial, 319
clear_trajectories, 320
correct_for_structure, 350
deactivate, 359
DevicePipelineEnableCCASDMIHandlerClass, 369
DevicePipelineEnableSACDMIHandlerClass, 370
DevicePipelineSetGROCHandlerClass, 371
DevicePipelineStatusHandlerClass, 372
disable_rho, 375
DMI_get_status, 379
DomeStatusHandlerClass, 406
enable_rho, 416
force_state_event, 525
ForceStatusEventsHandlerClass, 527
GenerateExceptionHandler, 532
get, 533
get_analysis_region, 534
get_correction, 539
get_current_local_sidereal_time, 544
get_current_pointing, 545
get_current_trajectory_info, 546
get_fiducial, 549
get_focus_gain, 550
get_focus_offset, 551
get_focus_window_length, 552
get_itf_offset, 557
get_metadata, 558
get_next_trajectory_info, 559
get_radec_offset, 560
get_sky_offset, 561
get_status, 562
get_tiptilt_gain, 563
get_tiptilt_offset, 564
get_tiptilt_window_length, 565
get_tuning, 566
go_next, 585
HelpHandler, 606
ImagePipelineGetExposureTimeHandlerClass, 639
ImagePipelineGetGainHandlerClass, 640
ImagePipelineGetOverScanHandlerClass, 641
ImagePipelineGetPhotPhotometryOptionsHandlerClass, 642
ImagePipelineGetRegionOfInterestHandlerClass, 643
ImagePipelineGetWFSPhotometryOptionsHandlerClass, 644
ImagePipelineSetCleanImagesHandlerClass, 645
ImagePipelineSetCleaningBiasTypeHandlerClass, 646
ImagePipelineSetFilterExposureTimeFactorHandlerClass, 647
ImagePipelineSetFilterSequenceHandlerClass, 648
ImagePipelineSetFilterZeroPointHandlerClass, 649
ImagePipelineSetObjectAndMagnitudesHandlerClass, 650
ImagePipelineSetOverScanHandlerClass, 651
ImagePipelineSetPhotometryOptionsHandlerClass, 652
ImagePipelineSetPhotPhotometryOptionsHandlerClass, 653

ImagePipelineSetProcessingTypeHandlerClass, 654
ImagePipelineSetWFSPhotometryOptionsHandlerClass, 655
ImagePipelineStartSendingImagesHandlerClass, 656
inject_metrology, 664
InterruptHandler, 666
load_trajectory, 698
LogRelayDriver, 709
MessageHandler, 734
MonitorDriver, 757
namedHandler, 785
NOOPHandler, 791
offset_fiducial, 797
offset_focus, 798
offset_itf, 799
offset_probe, 800
offset_sky, 801
offset_tiptilt, 802
offset_trajectory, 803
P, 812
pfip_clear_positions, 840
pfip_close_shutter, 841
pfip_insert_acquisition_pickoff, 845
pfip_move_probes, 847
pfip_open_shutter, 848
pfip_withdraw_acquisition_pickoff, 849
PipelineCyclePipelineHandlerClass, 888
PipelineRelinquishControlHandlerClass, 890
PipelineSetCycleTimeHandlerClass, 891
PipelineSetIntervalSpeedHandlerClass, 892
PipelineStartPipelineHandlerClass, 893
PipelineTakeControlHandlerClass, 895
PlotterDriver, 902
ProxyHandler, 970
query_trajectory, 987
reset, 1000
reset_focus_offset, 1003
reset_itf_offset, 1004
reset_radec_offset, 1007
reset_sky_offset, 1008
reset_tiptilt_offset, 1011
SampleHandlerClass, 1018
ServerHandler, 1027
set, 1028
set_analysis_region, 1029
set_fiducial, 1040
set_filter, 1041
set_focus, 1042
set_focus_gain, 1043
set_focus_window_length, 1044
set_metadata, 1045
set_position, 1046
set_tiptilt_gain, 1047
set_tiptilt_window_length, 1048
ShutterSetPowerHandlerClass, 1105
ShutterStatusHandlerClass, 1107
start, 1126
StatusHandler, 1162
stop, 1165
StructureMoveToCCASHandlerClass, 1179
StructureMoveToStructAzHandlerClass, 1180
StructureMoveToTrueAzHandlerClass, 1181
StructureOnLineHandlerClass, 1182
StructureStatusHandlerClass, 1184
StructureStowHandlerClass, 1185
SubSystemHandler, 1194
TapHandler, 1202
test, 1256
TimeoutHandler, 1272
toggle, 1280
toggle_gui_token, 1281
tracker_abort, 1285
tracker_handlers::abortMove, 208
tracker_handlers::clearError, 322
tracker_handlers::initTMCS, 663
tracker_handlers::moveRTF, 771
tracker_handlers::moveTSF, 776
tracker_handlers::setBufferSize, 1054
tracker_handlers::setTraj, 1086
tracker_handlers::timeToIntercept, 1275
tracker_init, 1287
tracker_move_angular, 1288
tracker_move_itf, 1289
tracker_move_tsf, 1290
tracker_position, 1291
tune, 1315
UnhandledHandler, 1409
unset_position, 1412
update_setup, 1413
UptimeHandler, 1414
wait_for_setup, 1438
WeatherGetDataHandlerClass, 1451
WeatherOnLineHandlerClass, 1453
ProcessorConfig, 959
set_options, 960
ProcessVariable, 961
productionTabUi, 964
program
TCSConfig, 1212
Proxy, 967
proxy, 966
ProxyConfig, 968
print_summary, 968
set_options, 968
ProxyException, 969
ProxyHandler, 970

process, 970
PSFFitData, 971
pt_itf, 182
find_radec, 182
inEllipse, 182
pthread_trampoline
tthread, 189
PupilData, 972
pytcs::tcs_event::TCSEvent, 1223
__init__, 1223
connect, 1223
next, 1223
subscribe, 1223
pytcs::tcsdb::attr_descriptor, 262
pytcs::tcsdb::key_descriptor, 673
pytcs::tcsdb::source_descriptor, 1117
pytcs::tcsdb::system_descriptor, 1196
pytcs::tcsdb::tcsdb, 1217
pytcs::TCSLog::TCSLog, 1241
__init__, 1241
pytcs::TCSNamed::TCSNamed, 1242
pytcs::tcssubsystem::TCSSubSystem, 1248

QBlinkingLineEdit, 973
QBlinkingLineEditPlugin, 974
QGraphicsCircleItem, 975
QLedIndicator, 976
QLedIndicatorPlugin, 978
QSparkLine, 979
QSparkLinePlugin, 980
QTcsFrame, 981
QTcsFramePlugin, 982
QTcsProbeAssembly, 983
quadFit, 985
Quaternion, 986
query_trajectory, 987
process, 987

radians_to_hours
hetlib::hetutils, 172
hetutils, 179
read_now, 990
ReadRegister, 991
ReadStatusValues
ProbeAssembly, 942
receive
TCSSocket, 1246
Receiver, 992
Receiver, 992
registerHandler, 992
start, 992
stop, 993
reg_id
mbintf_hldreg_bitfield_t, 721
mbintf_inpreg_analog_t, 722
mbintf_inpreg_bitfield_t, 724
sockcmd_t, 1112
registerHandler
Receiver, 992
regval_div
mbintf_inpreg_analog_t, 722
regval_mult
mbintf_inpreg_analog_t, 722
RelinquishControl, 997
Remote signalling, 93
remote_signal
MESSAGE_SETJMP, 93
replay::EventBuffer, 439
ReplyCurrentExposure
FCU_SubSystem, 473
ReplyLastExposure
FCU_SubSystem, 473
Reset, 998
reset, 1000
process, 1000
reset_clock_voltages, 1002
reset_focus_offset, 1003
process, 1003
reset_itf_offset, 1004
process, 1004
reset_mux, 1006
reset_radec_offset, 1007
process, 1007
reset_radec_offset_register
TCSDriver, 1219
reset_sky_offset, 1008
process, 1008
reset_tcs, 1010
CAMRADriver, 301
reset_tiptilt_offset, 1011
process, 1011
resetGuideRect
tcs_photometry::Photometry, 877
respond
Message, 732
RetractACQCameraMirror, 1012
RetractCWFSCameraMirror, 1013
rotateXY
MountModel, 761
RowDispatcher, 1015
rtf_coordinate, 1016
rtf_to_itf
MountModel, 761
run
tthread::ThreadWorker, 1270

Sample Servers Access System (Sample), 145
sampleConfig, 1017

print_summary, 1017
set_options, 1017
SampleHandlerClass, 1018
process, 1018
sampleServer, 1019
SAMSData, 1021
SAMSData::status, 1137
send
Sender, 1022
send_async
Sender, 1023
Sender, 1022
send, 1022
send_async, 1023
Sender, 1022
wait_for, 1023
SendExternalLogMessage
Logger, 705
SendLogMessage
Logger, 705
ser7_table_entry, 1025
Server, 1026
ServerHandler, 1027
process, 1027
set, 1028
process, 1028
set_analysis_region, 1029
process, 1029
set_bias_offset, 1031
set_ccd_temp, 1033
set_ccd_temp_xfer, 1035
set_clock_voltages, 1037
set_cryo_temp_xfer, 1038
set_data
Payload, 836
set_default
EventHandler, 441
set_fiducial, 1040
process, 1040
set_filter, 1041
EventSink, 449
process, 1041
set_focus, 1042
process, 1042
set_focus_gain, 1043
process, 1043
set_focus_window_length, 1044
process, 1044
set_global_defaults
EventHandler, 441
set_metadata, 1045
process, 1045
set_options
apcCmdConfig, 244
apcConfig, 245
CameraConfig, 277
CAMRAConfig, 298
CleanerConfig, 318
cmdConfig, 328
consoleConfig, 347
ControllerConfig, 348
DeviceConfig, 368
DomeConfig, 404
EPICSRelayConfig, 434
eventsConfig, 448
GroupConfig, 587
guiConfig, 589
IglooSensorsConfig, 624
legacyConfig, 675
LightSourceConfig, 696
namedConfig, 783
pasConfig, 816
pduConfig, 838
PFIPConfig, 854
PipelineConfig, 887
pointingConfig, 914
ProcessorConfig, 960
ProxyConfig, 968
sampleConfig, 1017
ShutterConfig, 1103
SpectrographConfig, 1122
StructureConfig, 1177
SubSystemConfig, 1192
TCSConfig, 1211
TCSServerConfig, 1244
TrackerConfig, 1294
WeatherConfig, 1447
WriteOneWireConfig, 1472
set_parm
Payload, 837
set_period
tthread::ThreadWorker, 1270
set_position, 1046
process, 1046
set_proportional
PIDLoop, 884
set_tiptilt_gain, 1047
process, 1047
set_tiptilt_window_length, 1048
process, 1048
set_trampoline
tthread::thread, 1261
tthread::ThreadWorker, 1270
SetACQCameraFilter, 1049
ACQ_Camera, 212
ACQ_SubSystem, 220
SetACQCameraFocus, 1050
ACQ_SubSystem, 221

SetACQCameraFocus_mm
ACQ_SubSystem, 221
SetACQCameraStatus
ACQ_Camera, 212
SetAnalysisRegion, 1051
SetBinning, 1052
setBufferSize, 1053
SetCleanImages, 1055
SetCleaningBiasType, 1056
SetCWFSCameraDeployed
ACQ_SubSystem, 221
SetCycleTime, 1057
SetDirection, 1058
SetDither
ACQ_SubSystem, 221
SetENWAFilter, 1059
SetENWAFilterPosition
ACQ_SubSystem, 221
SetExposuresPerCycle, 1060
SetExposureTime, 1061
SetFilterExposureTimeFactor, 1062
SetFilterSequence, 1063
SetFilterZeroPoint, 1064
SetGROC, 1065
SetGuideCameraFilter, 1066
setGuideRect
tcs_photometry::Photometry, 877
SetImageType, 1067
SetInt
EPICSComm, 430
SetIntervalSpeed, 1068
SetLightSourcePower, 1069
SetLightSourcePowerHoldTime, 1070
SetLightSourcePowerLevel, 1071
SetNumFlushes, 1072
SetObjectAndMagnitudes, 1073
SetOverScan, 1074
SetPhotPhotometryOptions, 1075
SetPixelFormat, 1076
setPixscale
tcs_photometry::Photometry, 877
SetPower, 1077
SetProbeGains, 1078
SetProcessImages, 1079
SetProcessingType, 1080
SetReadoutSpeed, 1081
SetRegionOfInterest, 1082
SetShutterStatus
ACQ_Shutter, 215
SetStoreImages, 1083
SetTargetTemperature, 1084
setTraj, 1085
SetValue
FCU_SubSystem, 474
setValues
EPICSComm, 430
MODBUSComm, 753
SetVelocity, 1087
SetWFSPhotometryOptions, 1088
SHGrid, 1089
ShotPoint, 1090
SHTrackerData, 1091
Shutter, 105, 106, 1092
shutter_Abort, 1093
shutter_Close, 1094
shutter_IssuePMACCmd, 1095
shutter_OnLine, 1096
shutter_Open, 1097
shutter_Reset, 1098
shutter_SetPower, 1099
shutter_Status, 1100
shutter_Stop, 1102
ShutterConfig, 1103
set_options, 1103
ShutterInfo, 1104
ShutterSetPowerHandlerClass, 1105
process, 1105
ShutterStatus, 1106
ShutterStatus::status, 1150
ShutterStatusHandlerClass, 1107
process, 1107
shutterTabUi, 1109
Simulated Camera, 115
size
Payload, 837
sleep_for
tthread::this_thread, 191
SlopeMatrix, 1110
sockcmd_reg_id_t, 1111
id, 1111
is_hld, 1111
sockcmd_t, 1112
cmd_code, 1112
reg_id, 1112
value, 1112
Socket, 1113
SocketException, 1114
sortSpect
hetlib::spectrograph::SpectrographList, 1123
spectrograph_id
SpectrographConfig, 1122
SpectrographConfig, 1121
print_summary, 1122
set_options, 1122
spectrograph_id, 1122
SphereRadiansToEncoderCounts
ProbePosition, 955
square_annulus_mask

tcs_photometry::Photometry, 877
st3, 184
getCurrentIndexTime, 184
start, 1126
process, 1126
Receiver, 992
start_clients
hetlib::tcsutils, 176
tcsutils, 186
StartPipeline, 1127
StartSendingImages, 1128
stash_temps
CAMRADriver, 301
StaticConfig, 1129
Status, 1135
StatusBlock, 1160
StatusData, 1161
StatusHandler, 1162
process, 1162
stencil, 1163
Stop, 1164
stop, 1165
process, 1165
Receiver, 993
tthread::ThreadWorker, 1270
StopPipeline, 1166
StopSendingImages, 1167
Stow, 1168
Structure, 99, 107, 108
Structure Mount Model, 91
structure_MoveToCCAS, 1169
structure_MoveToStructAz, 1170
structure_MoveToTrueAz, 1171
structure_OnLine, 1172
structure_Status, 1173
structure_Stop, 1174
structure_Stow, 1175
StructureClient, 1176
StructureConfig, 1177
set_options, 1177
StructureModel, 1178
StructureMoveToCCASHandlerClass, 1179
process, 1179
StructureMoveToStructAzHandlerClass, 1180
process, 1180
StructureMoveToTrueAzHandlerClass, 1181
process, 1181
StructureOnLineHandlerClass, 1182
process, 1182
StructureStatus, 1183
StructureStatus::status, 1156
StructureStatusHandlerClass, 1184
process, 1184
StructureStowHandlerClass, 1185
process, 1185
structureTabUi, 1187
subscribe
pytcs::tcs_event::TCSEvent, 1223
tcs_event::TCSEvent, 1222
SubSystem, 1188
subSystem, 1190
SubSystemConfig, 1192
print_summary, 1192
set_options, 1192
SubSystemHandler, 1194
process, 1194
subtract_background_fit
tcs_photometry::Photometry, 877
subtract_background_fit_annulus
tcs_photometry::Photometry, 877
subtract_background_fit_outer_edge
tcs_photometry::Photometry, 878

TakeControl, 1199
TakeExposure, 1200
ACQ_SubSystem, 221
Tap, 1201
TapHandler, 1202
process, 1202
TCS Application Class, 83
TCS Base Class, 84
TCS Base Configuration, 86
TCS Fixed precision time, 81
TCS SubSystem Class, 88
tcs_event::TCSEvent, 1222
__init__, 1222
connect, 1222
next, 1222
subscribe, 1222
tcs_photometry::AperturePhotResults, 254
tcs_photometry::ConfigureParams, 339
tcs_photometry::FitPars, 513
tcs_photometry::GuideRect, 594
tcs_photometry::GuiderMetrologyData, 596
tcs_photometry::Object, 792
tcs_photometry::PhotData, 872
tcs_photometry::Photometry, 873
aperture_phot, 875
assemble, 875
check_variance, 875
copy_square, 875
elliptical_annulus_mask, 875
elliptical_mask, 875
fast_aperture_phot, 875
fast_fixed_aperture_phot, 876
fast_kron_aperture_phot, 876
find_stars, 876
fit_bivariate_gaussian, 876

fit_bivariate_moffat, 876
fixed_aperture_phot, 876
Guider_photometry, 876
integrate_soft_e_mask_fixed, 876
kron_aperture_phot, 877
mark_pixels, 877
resetGuideRect, 877
setGuideRect, 877
setPixscale, 877
square_annulus_mask, 877
subtract_background_fit, 877
subtract_background_fit_annulus, 877
subtract_background_fit_outer_edge, 878
TTCAM_photometry, 878
tcs_photometry::Pixel, 896
tcs_photometry::TTCAMMetrologyData, 1314
TCSApp, 1203
TCSBaseClass, 1204
TCSBaseException, 1205
TCSClient, 1206
TCSClient::TelescopeData, 1254
tcsCmd, 1207
TCSConfig, 1208
config_root, 1211
configure, 1210
debug, 1211
help, 1211
hostname, 1211
log_levels, 1211
name, 1212
print_summary, 1211
program, 1212
set_options, 1211
TCSConfig, 1210
version, 1212
tcsconfig, 185
TCSConfig::mapEntry, 718
TCSConfigException, 1213
tcsConsole, 1214
tcsdb::attr_descriptor, 261
tcsdb::key_descriptor, 672
tcsdb::source_descriptor, 1116
tcsdb::system_descriptor, 1197
tcsdb::tcsdb, 1216
TCSDriver, 1218
cancel_trajectory, 1219
clear_trajectories, 1219
get_radec_offset_register, 1219
go_next, 1219
reset_radec_offset_register, 1219
trajectory, 1219
trajectory_info, 1219
TCSEvent::TCSEvent, 1221
tcsEvents, 1224
TCSException, 1225
TcsException, 1226
tcsGui, 1227
tcsGui::soundObject, 1115
tcsGuiThreadWorker, 1233
TCSHandler, 1239
TCSLog::TCSLog, 1240
__init__, 1240
TCSNamed::TCSNamed, 1243
TCSServerConfig, 1244
print_summary, 1244
set_options, 1244
TCSSocket, 1245
connect, 1246
connected, 1246
disconnect, 1246
flush, 1246
receive, 1246
transmit, 1246
TCSSubSystem, 1249
tcssubsystem::TCSSubSystem, 1250
TCSTelnet, 1252
tcsutils, 186
getposition, 186
moveStructure, 186
named_lookup, 186
start_clients, 186
Tde
hetlib::hetutils, 172
hetutils, 179
telecen2probe
pfip_coord, 842
telescope, 1253
Telescope Control System (TCS), 151
Test, 1255
test, 1256
process, 1256
ThermoCubeStatus, 1257
ThermoCubeStatus::status, 1148
thread
tthread::thread, 1259, 1260
thread_execute_once
tthread::thread, 1261
thread_execute_periodic
tthread::ThreadWorker, 1270
thread_local
tinythread.h, 1477
threadsafe_deque, 1265
threadsafe_list, 1266
threadsafe_queue, 1267
ThreadWorker
tthread::ThreadWorker, 1269
TimeoutHandler, 1272
process, 1272

TimeoutNextMove, 1273
timeToIntercept, 1276
TinyThread++ API Reference, 95
tinythread.h
    _TTHREAD_DISABLE_ASSIGNMENT, 1477
    thread_local, 1477
Tip Tilt Light Source, 131
tle_t, 1277
TMCSState, 1278
toggle, 1280
    process, 1280
toggle_gui_token, 1281
    process, 1281
TopConfig, 1282
TopDriver, 1283
tracCoord, 1284
Tracker Motion Control System (TMCS) API, 152
Tracker Server, 168
tracker_abort, 1285
    process, 1285
tracker_client
    TrackerDriver, 1296
tracker_coordinate, 1286
tracker_handlers::abortMove, 208
    process, 208
tracker_handlers::clearError, 322
    process, 322
tracker_handlers::initTMCS, 663
    process, 663
tracker_handlers::moveRTF, 771
    process, 771
tracker_handlers::moveTSF, 776
    process, 776
tracker_handlers::setBufferSize, 1054
    process, 1054
tracker_handlers::setTraj, 1086
    process, 1086
tracker_handlers::timeToIntercept, 1275
    process, 1275
tracker_init, 1287
    process, 1287
tracker_move_angular, 1288
    process, 1288
tracker_move_itf, 1289
    process, 1289
tracker_move_tsf, 1290
    process, 1290
tracker_position, 1291
    process, 1291
tracker_server
    TrackerDriver, 1296
TrackerClient, 1293
TrackerConfig, 1294
    print_summary, 1294
    set_options, 1294
TrackerData, 1295
TrackerDriver, 1296
    tracker_client, 1296
    tracker_server, 1296
TrackerException, 1297
TrackerHandler, 1298
TrackerPosition, 1300
TrackerPosition::status, 1151
TrackerStatus, 1301
TrackerStatus::status, 1141
trackerStatusTabUi, 1303
trackerTabUi, 1306
tracks_
    Point, 909
traj_class::Trajectory, 1307
    __init__, 1307
traj_class::TrajectoryList, 1310
Trajectory, 1308
trajectory
    TCSDriver, 1219
trajectory_info
    TCSDriver, 1219
TrajectoryException, 1309
trajectoryTabUi, 1312
transmit
    TCSSocket, 1246
try_lock
    tthread::fast_mutex, 462
    tthread::mutex, 778
    tthread::recursive_mutex, 994
tsf_coordinate, 1313
TTCAM_photometry
    tcs_photometry::Photometry, 878
tthread, 188
    _threadpool_start_thread, 189
    pthread_trampoline, 189
tthread::_thread_start_functor, 199
tthread::_thread_start_functor_arg, 200
tthread::_thread_start_info, 201
tthread::_thread_start_member, 202
tthread::_thread_start_member_arg, 203
tthread::chrono, 190
tthread::chrono::duration, 410
tthread::condition_variable, 335
    notify_all, 335
    notify_one, 335
    wait, 336
    wait_for, 336
tthread::fast_mutex, 462
    lock, 462
    try_lock, 462
    unlock, 462

tthread::lock_guard, 702
tthread::mutex, 778
    lock, 778
    try_lock, 778
    unlock, 778
tthread::ratio, 988
tthread::recursive_mutex, 994
    lock, 994
    try_lock, 994
    unlock, 994
tthread::this_thread, 191
    sleep_for, 191
    yield, 191
tthread::thread, 1258
    ∼thread, 1260
    detach, 1260
    hardware_concurrency, 1260
    join, 1261
    joinable, 1261
    native_handle, 1261
    set_trampoline, 1261
    thread, 1259, 1260
    thread_execute_once, 1261
tthread::thread::id, 620
    id, 620
tthread::ThreadPool, 1262
tthread::ThreadPoolException, 1263
tthread::ThreadPoolWorker, 1264
tthread::ThreadWorker, 1268
    is_running, 1270
    run, 1270
    set_period, 1270
    set_trampoline, 1270
    stop, 1270
    thread_execute_periodic, 1270
    ThreadWorker, 1269
tthread::Timer, 1274
tune, 1315
    process, 1315
turn_off_leds, 1316

Ui::aboutDialog, 209
Ui::acqCamStatusTab, 224
Ui::alertsDialog, 236
Ui::apcTab, 251
Ui::cameraControlTab, 278
Ui::cameraImageTab, 286
Ui::cameraMetrologyControlTab, 291
Ui::cameraMetrologyDataTab, 293
Ui::configureSystemsDialog, 341
Ui::configureTabsDialog, 344
Ui::confirmOffsetDialog, 346
Ui::dmiTab, 385
Ui::domeTab, 407
Ui::editPreferencesDialog, 413
Ui::eventMonitorTab, 445
Ui::fcuControlTab, 478
Ui::fcuStatusTab, 503
Ui::fiducialDialog, 507
Ui::focalPlaneShutterTab, 521
Ui::guideBoxDialog, 592
Ui::iglooTab, 631
Ui::legacyTab, 690
Ui::loadMarkersDialog, 701
Ui::mainWindow, 717
Ui::pfipControlTab, 855
Ui::pfipEnvironmentTab, 858
Ui::photometryOptionsDialog, 880
Ui::probeAssemblyTab, 945
Ui::probeControlTab, 947
Ui::probeGainControlTab, 950
Ui::probeStatusTab, 957
Ui::productionTab, 963
Ui::shutterTab, 1108
Ui::structureTab, 1186
Ui::trackerStatusTab, 1302
Ui::trackerTab, 1305
Ui::trajectoryTab, 1311
Ui::videoSwitchTab, 1427
Ui::virusEnvironmentTab, 1432
Ui::weatherDataTab, 1449
Ui::webBrowserTab, 1454
Ui_aboutDialog, 1318
Ui_acqCamStatusTab, 1319
Ui_alertsDialog, 1321
Ui_apcTab, 1322
Ui_cameraControlTab, 1323
Ui_cameraImageTab, 1325
Ui_cameraMetrologyControlTab, 1327
Ui_cameraMetrologyDataTab, 1328
Ui_configureSystemsDialog, 1329
Ui_configureTabsDialog, 1330
Ui_confirmOffsetDialog, 1331
Ui_dmiTab, 1332
Ui_domeTab, 1334
Ui_editPreferencesDialog, 1336
Ui_eventMonitorTab, 1338
Ui_fcuControlTab, 1339
Ui_fcuStatusTab, 1341
Ui_fiducialDialog, 1343
Ui_focalPlaneShutterTab, 1344
Ui_guideBoxDialog, 1345
Ui_iglooTab, 1346
Ui_legacyTab, 1347
Ui_loadMarkersDialog, 1351
Ui_mainWindow, 1352
Ui_pfipControlTab, 1354
Ui_pfipEnvironmentTab, 1356

Ui_photometryOptionsDialog, 1358
Ui_probeAssemblyTab, 1359
Ui_probeControlTab, 1360
Ui_probeGainControlTab, 1364
Ui_probeStatusTab, 1365
Ui_productionTab, 1369
Ui_shutterTab, 1377
Ui_structureTab, 1378
Ui_trackerStatusTab, 1379
Ui_trackerTab, 1387
Ui_trajectoryTab, 1394
Ui_videoSwitchTab, 1398
Ui_virusEnvironmentTab, 1399
Ui_weatherDataTab, 1405
Ui_webBrowserTab, 1408
UnhandledHandler, 1409
    process, 1409
Unimplemented, 1410
UnknownException, 1411
unlock
    tthread::fast_mutex, 462
    tthread::mutex, 778
    tthread::recursive_mutex, 994
unset_position, 1412
    process, 1412
update_setup, 1413
    process, 1413
UpdateProbePositions
    ProbeAssembly, 942
UptimeHandler, 1414
    process, 1414
utc, 1415
utc_iso8601, 1416
util::NamedReader, 786
utilityGui, 1417
utilityGuiThreadWorker, 1419

validPixelFormat, 1420
value
    EventHandler, 442
    sockcmd_t, 1112
variable
    Event, 437
Venc1PrintRegister, 1422
Venc1WriteRegister, 1423
Venc2PrintRegister, 1424
Venc2WriteRegister, 1425
VENCHandler, 1426
version
    TCSConfig, 1212
videoSwitchTabUi, 1428
videoSwitchTabUiThreadWorker, 1430
VIRUS_Monitor, 1431
virusEnvironmentTabUi, 1433
virusEnvironmentTabUi::virusTempData, 1435
VirusMonitorStatus, 1434
VirusMonitorStatus::alarms, 234
VirusMonitorStatus::status, 1153

wait
    EventHandler, 442
    tthread::condition_variable, 336
wait_for
    Sender, 1023
    tthread::condition_variable, 336
wait_for_readout, 1437
wait_for_setup, 1438
    process, 1438
wait_for_shutter_close, 1440
wait_for_write, 1442
WaitForProbeEnabled
    PFIPProbeInfo, 867
Wave Front Sensor Image Processor, 126
Weather, 100, 109, 110
weather, 1443
weather_GetData, 1444
weather_OnLine, 1445
WeatherClient, 1446
WeatherConfig, 1447
    set_options, 1447
WeatherData, 1448
WeatherData::statistic, 1132
WeatherData::status, 1136
weatherDataTabUi, 1450
WeatherGetDataHandlerClass, 1451
    process, 1451
WeatherOnLine, 1452
WeatherOnLineHandlerClass, 1453
    process, 1453
webBrowserTabUi, 1455
webBrowserTabUiThreadWorker, 1456
wfs_photometry::ConfigureParams, 340
wfs_photometry::WFSCentroid, 1457
wfs_photometry::WFSCentroid::object_t, 794
wfs_photometry::WFSCentroid::refList_t, 996
wfs_photometry::WFSCentroid::vector_t, 1421
wfs_photometry::WFSFit, 1460
wfs_photometry::WFSFitException, 1461
wfsData, 1458
WFSException, 1459
WFSHandler, 1462
wfslib, 1463
wfslib_het, 1465
wfsLogger, 1468
WFSLoop, 1469
WFSToTCSLogLevelMap_t, 1470
WhichMotor
    probeassembly, 142

WriteOneWire, 1471
WriteOneWireConfig, 1472
    print_summary, 1472
    set_options, 1472
WriteRegister, 1473
WriteShutterOpenClose
    ACQ_Shutter, 216

xsky_find, 1474

yield
    tthread::this_thread, 191