

KatanaNativeInterface Reference Manual
\$VERSION\$

Generated by Doxygen 1.5.1

Thu Sep 27 08:29:49 2007

Contents

1	"Katana Native Interface Documentation"	1
2	KatanaNativeInterface Module Index	3
2.1	KatanaNativeInterface Modules	3
3	KatanaNativeInterface Directory Hierarchy	5
3.1	KatanaNativeInterface Directories	5
4	KatanaNativeInterface Namespace Index	7
4.1	KatanaNativeInterface Namespace List	7
5	KatanaNativeInterface Hierarchical Index	9
5.1	KatanaNativeInterface Class Hierarchy	9
6	KatanaNativeInterface Class Index	13
6.1	KatanaNativeInterface Class List	13
7	KatanaNativeInterface File Index	17
7.1	KatanaNativeInterface File List	17
8	KatanaNativeInterface Module Documentation	19
8.1	Exceptions	19
9	KatanaNativeInterface Directory Documentation	21
9.1	include/common/ Directory Reference	21
9.2	include/ Directory Reference	22
9.3	include/KNI/ Directory Reference	23
9.4	include/KNI_InvKin/ Directory Reference	24
9.5	include/KNI_LM/ Directory Reference	25
10	KatanaNativeInterface Namespace Documentation	27
10.1	KNI Namespace Reference	27

10.2 KNI_MHF Namespace Reference	29
11 KatanaNativeInterface Class Documentation	33
11.1 CannotGetSetPortAttributesException Class Reference	33
11.2 CannotOpenPortException Class Reference	35
11.3 CCdlBase Class Reference	37
11.4 CCdlCOM Class Reference	39
11.5 CCdlSocket Class Reference	43
11.6 CCplBase Class Reference	47
11.7 CCplSerial Class Reference	50
11.8 CCplSerialCRC Class Reference	53
11.9 CikBase Class Reference	57
11.10Katana Class Reference	63
11.11CKatBase Class Reference	76
11.12CLMBase Class Reference	86
11.13CMotBase Class Reference	94
11.14ConfigFileEntryNotFoundExcepion Class Reference	107
11.15ConfigFileOpenException Class Reference	109
11.16ConfigFileSectionNotFoundExcepion Class Reference	111
11.17ConfigFileStateException Class Reference	113
11.18ConfigFileSubsectionNotFoundExcepion Class Reference	115
11.19ConfigFileSyntaxErrorException Class Reference	117
11.20Context Struct Reference	119
11.21CSctBase Class Reference	120
11.22DeviceReadException Class Reference	123
11.23DeviceWriteException Class Reference	125
11.24ErrorException Class Reference	127
11.25Exception Class Reference	129
11.26JointSpeedException Class Reference	131
11.27KNI::KatanaKinematics Class Reference	133
11.28KNI::KatanaKinematics5M180 Class Reference	136
11.29KNI::KatanaKinematics5M180::angles_calc Struct Reference	140
11.30KNI::KatanaKinematics5M180::position Struct Reference	142
11.31KNI::KatanaKinematics6M180 Class Reference	143
11.32KNI::KatanaKinematics6M180::angles_calc Struct Reference	147
11.33KNI::KatanaKinematics6M180::position Struct Reference	149
11.34KNI::KatanaKinematics6M90G Class Reference	150

11.35KNI::KatanaKinematics6M90G::angles_calc Struct Reference	154
11.36KNI::KatanaKinematics6M90G::position Struct Reference	156
11.37KNI::KatanaKinematics6M90T Class Reference	157
11.38KNI::KatanaKinematics6M90T::angles_calc Struct Reference	162
11.39KNI::KatanaKinematics6M90T::position Struct Reference	164
11.40KNI::KinematicParameters Struct Reference	165
11.41KNI::KinematicsDefaultEncMinAlgorithm Struct Reference	166
11.42KNI::kmlFactory Class Reference	167
11.43MotorCrashException Class Reference	170
11.44MotorOutOfRangeException Class Reference	172
11.45MotorTimeoutException Class Reference	174
11.46KNI::NoSolutionException Class Reference	176
11.47ParameterReadingException Class Reference	178
11.48ParameterWritingException Class Reference	180
11.49PortNotOpenException Class Reference	182
11.50ReadNotCompleteException Class Reference	184
11.51ReadWriteNotCompleteException Class Reference	187
11.52SlaveErrorException Class Reference	189
11.53TBlendtrace Struct Reference	191
11.54TBLENDtrajectory Struct Reference	194
11.55TCdlCOMDesc Struct Reference	196
11.56THeader Struct Reference	198
11.57KNI::Timer Class Reference	199
11.58TKatCBX Struct Reference	201
11.59TKatCTB Struct Reference	202
11.60TKatECH Struct Reference	203
11.61TKatEFF Struct Reference	204
11.62TKatGNL Struct Reference	205
11.63TKatIDS Struct Reference	206
11.64TKatMFW Struct Reference	207
11.65TKatMOT Struct Reference	208
11.66TKatSCT Struct Reference	210
11.67TLM_points Struct Reference	212
11.68TLMtrajectory Struct Reference	213
11.69TMLMIP Struct Reference	216
11.70TMotAPS Struct Reference	217

11.71	TMotCLB Struct Reference	218
11.72	TMotDesc Struct Reference	220
11.73	TMotDYL Struct Reference	221
11.74	TMotENL Struct Reference	224
11.75	TMotGNL Struct Reference	226
11.76	TMotInit Struct Reference	228
11.77	TMotPVP Struct Reference	230
11.78	TMotSCP Struct Reference	232
11.79	TMotSFW Struct Reference	236
11.80	TMotTPS Struct Reference	238
11.81	TPacket Struct Reference	239
11.82	TPoint3D Struct Reference	240
11.83	TPoint6D Struct Reference	241
11.84	TSctDAT Struct Reference	243
11.85	TSctDesc Struct Reference	244
11.86	TSctGNL Struct Reference	245
11.87	TSplinepoint Struct Reference	247
11.88	KNI_MHF::unary_deg2rad<_T> Struct Template Reference	248
11.89	KNI_MHF::unary_precalc_cos<_T> Struct Template Reference	249
11.90	KNI_MHF::unary_precalc_sin<_T> Struct Template Reference	250
11.91	KNI_MHF::unary_rad2deg<_T> Struct Template Reference	251
11.92	WaitParameterException Class Reference	252
11.93	WriteNotCompleteException Class Reference	254
11.94	WrongCRCEXception Class Reference	257
11.95	WrongParameterException Class Reference	259
12	KatanaNativeInterface File Documentation	261
12.1	include/common/dllexport.h File Reference	261
12.2	include/common/exception.h File Reference	263
12.3	include/common/MathHelperFunctions.h File Reference	264
12.4	include/common/Timer.h File Reference	266
12.5	include/KNI/cdlBase.h File Reference	267
12.6	include/KNI/cdlCOM.h File Reference	268
12.7	include/KNI/cdlCOMExceptions.h File Reference	269
12.8	include/KNI/cdlSocket.h File Reference	271
12.9	include/KNI/cplBase.h File Reference	272
12.10	include/KNI/cplSerial.h File Reference	274

12.11	include/KNI/CRC.h File Reference	276
12.12	include/KNI/kmlBase.h File Reference	277
12.13	include/KNI/kmlCommon.h File Reference	280
12.14	include/KNI/kmlExt.h File Reference	282
12.15	include/KNI/kmlFactories.h File Reference	283
12.16	include/KNI/kmlMotBase.h File Reference	284
12.17	include/KNI/kmlSctBase.h File Reference	287
12.18	include/KNI_InvKin/ikBase.h File Reference	288
12.19	include/KNI_InvKin/KatanaKinematics.h File Reference	289
12.20	include/KNI_InvKin/KatanaKinematics5M180.h File Reference	290
12.21	include/KNI_InvKin/KatanaKinematics6M180.h File Reference	291
12.22	include/KNI_InvKin/KatanaKinematics6M90G.h File Reference	292
12.23	include/KNI_InvKin/KatanaKinematics6M90T.h File Reference	293
12.24	include/KNI_InvKin/KatanaKinematicsDecisionAlgorithms.h File Reference	294
12.25	include/KNI_LM/lmBase.h File Reference	295
12.26	include/kniBase.h File Reference	296

Chapter 1

"Katana Native Interface Documentation"

Chapter 2

KatanaNativeInterface Module Index

2.1 KatanaNativeInterface Modules

Here is a list of all modules:

Exceptions	19
----------------------	----

Chapter 3

KatanaNativeInterface Directory Hierarchy

3.1 KatanaNativeInterface Directories

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

include	22
common	21
KNI	23
KNI_InvKin	24
KNI_LM	25

Chapter 4

KatanaNativeInterface Namespace Index

4.1 KatanaNativeInterface Namespace List

Here is a list of all namespaces with brief descriptions:

KNI	27
KNI_MHF	29

Chapter 5

KatanaNativeInterface Hierarchical Index

5.1 KatanaNativeInterface Class Hierarchy

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

CCdlBase	37
CCdlCOM	39
CCdlSocket	43
CCplBase	47
CCplSerial	50
CCplSerialCRC	53
CKatana	63
CikBase	57
CLMBase	86
CKatBase	76
CMotBase	94
Context	119
CSetBase	120
std::exception	
Exception	129
CannotGetSetPortAttributesException	33
CannotOpenPortException	35
ConfigFileEntryNotFoundException	107
ConfigFileOpenException	109
ConfigFileSectionNotFoundException	111
ConfigFileStateException	113
ConfigFileSubsectionNotFoundException	115
ConfigFileSyntaxErrorException	117
DeviceReadException	123
DeviceWriteException	125
ErrorException	127
JointSpeedException	131
KNI::NoSolutionException	176
MotorCrashException	170
MotorOutOfRangeException	172

MotorTimeoutException	174
ParameterReadingException	178
ParameterWritingException	180
PortNotOpenException	182
ReadWriteNotCompleteException	187
ReadNotCompleteException	184
WriteNotCompleteException	254
SlaveErrorException	189
WaitParameterException	252
WrongCRCEXception	257
WrongParameterException	259
KNI::KatanaKinematics	133
KNI::KatanaKinematics5M180	136
KNI::KatanaKinematics6M180	143
KNI::KatanaKinematics6M90G	150
KNI::KatanaKinematics6M90T	157
KNI::KatanaKinematics5M180::angles_calc	140
KNI::KatanaKinematics5M180::position	142
KNI::KatanaKinematics6M180::angles_calc	147
KNI::KatanaKinematics6M180::position	149
KNI::KatanaKinematics6M90G::angles_calc	154
KNI::KatanaKinematics6M90G::position	156
KNI::KatanaKinematics6M90T::angles_calc	162
KNI::KatanaKinematics6M90T::position	164
KNI::KinematicParameters	165
KNI::KinematicsDefaultEncMinAlgorithm	166
KNI::kmlFactory	167
TBlendtrace	191
TBLENDtrajectory	194
TCdlCOMDesc	196
THeader	198
KNI::Timer	199
TKatCBX	201
TKatCTB	202
TKatECH	203
TKatEFF	204
TKatGNL	205
TKatIDS	206
TKatMFW	207
TKatMOT	208
TKatSCT	210
TLM_points	212
TLMtrajectory	213
TMLMIP	216
TMotAPS	217
TMotCLB	218
TMotDesc	220
TMotDYL	221
TMotENL	224
TMotGNL	226
TMotInit	228
TMotPVP	230
TMotSCP	232

TMotSFW	236
TMotTPS	238
TPacket	239
TPoint3D	240
TPoint6D	241
TSctDAT	243
TSctDesc	244
TSctGNL	245
TSplinepoint	247
KNI_MHF::unary_deg2rad<_T>	248
KNI_MHF::unary_precalc_cos<_T>	249
KNI_MHF::unary_precalc_sin<_T>	250
KNI_MHF::unary_rad2deg<_T>	251

Chapter 6

KatanaNativeInterface Class Index

6.1 KatanaNativeInterface Class List

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

CannotGetSetPortAttributesException (Could not set or get the attributes for the given serial communication device)	33
CannotOpenPortException (Failed to open the serial communication device)	35
CCdlBase (Abstract base class for devices)	37
CCdlCOM (Encapsulates the serial port device)	39
CCdlSocket (Encapsulates the socket communication device)	43
CCplBase (Abstract base class for protocol definiton)	47
CCplSerial (Base class of two different serial protocols)	50
CCplSerialCRC (Implement the Serial-Zero protocol Initializing functionCommunication functionImplement the Serial-CRC protocol)	53
CikBase	57
CKatana (Extended Katana class with additional functions)	63
CKatBase (Base Katana class)	76
CLMBase (Linear movement Class)	86
CMotBase (Motor class)	94
ConfigFileEntryNotFoundException (The requested entry could not be found)	107
ConfigFileOpenException (Accessing the given configuration file failed (may be: access denied or wrong path))	109
ConfigFileSectionNotFoundException (The requested section could not be found)	111
ConfigFileStateException (The state of the configuration file wasn't "good")	113
ConfigFileSubsectionNotFoundException (The requested subsection could not be found)	115
ConfigFileSyntaxErrorException (There was a syntax error in the configuration file)	117
Context	119
CSctBase (Sensor Controller class)	120
DeviceReadException (Reading from the serial communication device failed)	123
DeviceWriteException (Writing to the serial communication device failed)	125
ErrorException (The Katana returned an error string)	127
Exception	129
JointSpeedException (Joint speed too high)	131
KNI::KatanaKinematics (The base class for all kinematic implementations)	133
KNI::KatanaKinematics5M180 (

Author:

Tiziano Mueller <tiziano.mueller@neuronics.ch>	
)	136
KNI::KatanaKinematics5M180::angles_calc	140
KNI::KatanaKinematics5M180::position	142
KNI::KatanaKinematics6M180	(

Author:

Tiziano Mueller <tiziano.mueller@neuronics.ch>	
)	143
KNI::KatanaKinematics6M180::angles_calc	147
KNI::KatanaKinematics6M180::position	149
KNI::KatanaKinematics6M90G	(

Author:

Tiziano Mueller <tiziano.mueller@neuronics.ch>	
)	150
KNI::KatanaKinematics6M90G::angles_calc	154
KNI::KatanaKinematics6M90G::position	156
KNI::KatanaKinematics6M90T	(

Author:

Tiziano Mueller <tiziano.mueller@neuronics.ch>	
)	157
KNI::KatanaKinematics6M90T::angles_calc	162
KNI::KatanaKinematics6M90T::position	164
KNI::KinematicParameters (To pass different parameters for the kinematic implementations)	165
KNI::KinematicsDefaultEncMinAlgorithm	166
KNI::kmlFactory (This class is for internal use only It may change at any time It shields the configuration file parsing)	167
MotorCrashException (The requested motor crashed during the movement)	170
MotorOutOfRangeException (The encoders for the given motor were out of range)	172
MotorTimeoutException (The timeout elapsed for the given motor and target position)	174
KNI::NoSolutionException (No solution found for the given cartesian coordinates)	176
ParameterReadingException (There was an error while reading a parameter from the robot)	178
ParameterWritingException (The data you wanted to send to the robot was invalid)	180
PortNotOpenException (The port was not open)	182
ReadNotCompleteException (The Katana didn't answer correctly within the given timeout)	184
ReadWriteNotCompleteException (This exception is the base for the WriteNotComplete and ReadNotCompleteException)	187
SlaveErrorException (Slave error occurred)	189
TBlendtrace	191
TBLENDtrajectory ([LMBLEND] Trajectory points)	194
TCdIComDesc (This structrue stores the attributes for a serial port device)	196
THeader (Header of a communication packet)	198
KNI::Timer (Provides a stop-watch-like class with a resolution of milliseconds)	199
TKatCBX ([CBX] connector box)	201
TKatCTB ([CTB] command table defined in the firmware)	202
TKatECH ([ECH] echo)	203
TKatEFF (Inverse Kinematics structure of the endeffektor)	204
TKatGNL ([GNL] general robot attributes)	205
TKatIDS ([IDS] identification string)	206
TKatMFW ([MFW] master firmware version/revision number)	207

TKatMOT ([MOT] every motor's attributes)	208
TKatSCT ([SCT] every sens ctrl's attributes)	210
TLM_points ([LM] linear movement: points to be interpolated)	212
TLMtrajectory ([LM] linear movement: parameters)	213
TMLMIP ([LM] Store intermediate targets for multiple linear movements)	216
TMotAPS ([APS] actual position)	217
TMotCLB (Calibration structure for single motors)	218
TMotDesc (Motor description (partly))	220
TMotDYL ([DYL] dynamic limits)	221
TMotENL ([ENL] limits in encoder values (INTERNAL STRUCTURE!))	224
TMotGNL ([GNL] motor generals)	226
TMotInit (Initial motor parameters)	228
TMotPVP ([PVP] position, velocity, pulse width modulation)	230
TMotSCP ([SCP] static controller parameters)	232
TMotSFW ([SFW] slave firmware)	236
TMotTPS ([TPS] target position)	238
TPacket (Communication packet)	239
TPoint3D	240
TPoint6D ([LMBLEND] Standard coordinates for a point in space)	241
TSctDAT ([DAT] sensor data)	243
TSctDesc (Sensor controller description (partly))	244
TSctGNL ([GNL] controller generals)	245
TSplinepoint	247
KNI_MHF::unary_deg2rad<_T> (Function-object version of rad2deg)	248
KNI_MHF::unary_precalc_cos<_T> (.	
See also:	
unary_precalc_sin	249
KNI_MHF::unary_precalc_sin<_T> (Function-object which calculates sinus for n-elements of a container if used together with a STL algorithm)	250
KNI_MHF::unary_rad2deg<_T> (Function-object version of rad2deg)	251
WaitParameterException (Wait parameter set to false)	252
WriteNotCompleteException (Not all bytes could be written to the serial communication device)	254
WrongCRCException (CRC check for the answer package failed)	257
WrongParameterException (The given parameter was wrong)	259

Chapter 7

KatanaNativeInterface File Index

7.1 KatanaNativeInterface File List

Here is a list of all files with brief descriptions:

include/kniBase.h	296
include/common/dllexport.h	261
include/common/exception.h	263
include/common/MathHelperFunctions.h	264
include/common/Timer.h	266
include/KNI/cdlBase.h	267
include/KNI/cdlCOM.h	268
include/KNI/cdlCOMExceptions.h	269
include/KNI/cdlSocket.h	271
include/KNI/cplBase.h	272
include/KNI/cplSerial.h	274
include/KNI/CRC.h	276
include/KNI/kmlBase.h	277
include/KNI/kmlCommon.h	280
include/KNI/kmlExt.h	282
include/KNI/kmlFactories.h	283
include/KNI/kmlMotBase.h	284
include/KNI/kmlSctBase.h	287
include/KNI_InvKin/ikBase.h	288
include/KNI_InvKin/KatanaKinematics.h	289
include/KNI_InvKin/KatanaKinematics5M180.h	290
include/KNI_InvKin/KatanaKinematics6M180.h	291
include/KNI_InvKin/KatanaKinematics6M90G.h	292
include/KNI_InvKin/KatanaKinematics6M90T.h	293
include/KNI_InvKin/KatanaKinematicsDecisionAlgorithms.h	294
include/KNI_LM/lmBase.h	295

Chapter 8

KatanaNativeInterface Module Documentation

8.1 Exceptions

Classes

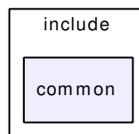
- struct [Context](#)
- class [Exception](#)
- class [CannotOpenPortException](#)
Failed to open the serial communication device.
- class [CannotGetSetPortAttributesException](#)
Could not set or get the attributes for the given serial communication device.
- class [PortNotOpenException](#)
The port was not open.
- class [DeviceReadException](#)
Reading from the serial communication device failed.
- class [DeviceWriteException](#)
Writing to the serial communication device failed.
- class [ReadWriteNotCompleteException](#)
This exception is the base for the [WriteNotComplete](#) and [ReadNotCompleteException](#).
- class [WriteNotCompleteException](#)
Not all bytes could be written to the serial communication device.
- class [ReadNotCompleteException](#)
The Katana didn't answer correctly within the given timeout.
- class [ErrorException](#)
The Katana returned an error string.

- class [WrongCRCException](#)
CRC check for the answer package failed.
- class [SlaveErrorException](#)
Slave error occurred.
- class [ParameterReadingException](#)
There was an error while reading a parameter from the robot.
- class [ParameterWritingException](#)
The data you wanted to send to the robot was invalid.
- class [WrongParameterException](#)
The given parameter was wrong.
- class [MotorOutOfRangeException](#)
The encoders for the given motor were out of range.
- class [MotorTimeoutException](#)
The timeout elapsed for the given motor and target position.
- class [MotorCrashException](#)
The requested motor crashed during the movement.
- class [ConfigFileOpenException](#)
Accessing the given configuration file failed (may be: access denied or wrong path).
- class [ConfigFileStateException](#)
The state of the configuration file wasn't "good".
- class [ConfigFileSectionNotFoundException](#)
The requested section could not be found.
- class [ConfigFileSubsectionNotFoundException](#)
The requested subsection could not be found.
- class [ConfigFileEntryNotFoundException](#)
The requested entry could not be found.
- class [ConfigFileSyntaxErrorException](#)
There was a syntax error in the configuration file.
- class [KNI::NoSolutionException](#)
No solution found for the given cartesian coordinates.
- class [JointSpeedException](#)
Joint speed too high.
- class [WaitParameterException](#)
Wait parameter set to false.

Chapter 9

KatanaNativeInterface Directory Documentation

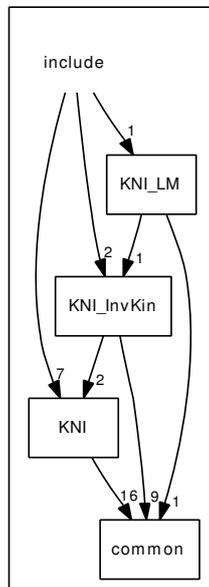
9.1 include/common/ Directory Reference



Files

- file [dllexport.h](#)
- file [exception.h](#)
- file [MathHelperFunctions.h](#)
- file [Timer.h](#)

9.2 include/ Directory Reference



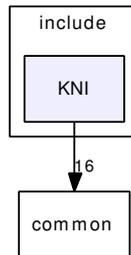
Directories

- directory [common](#)
- directory [KNI](#)
- directory [KNI_InvKin](#)
- directory [KNI_LM](#)

Files

- file [kniBase.h](#)

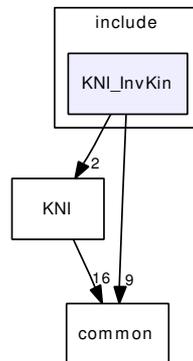
9.3 include/KNI/ Directory Reference



Files

- file [cdlBase.h](#)
- file [cdlCOM.h](#)
- file [cdlCOMExceptions.h](#)
- file [cdlSocket.h](#)
- file [cplBase.h](#)
- file [cplSerial.h](#)
- file [CRC.h](#)
- file [kmlBase.h](#)
- file [kmlCommon.h](#)
- file [kmlExt.h](#)
- file [kmlFactories.h](#)
- file [kmlMotBase.h](#)
- file [kmlSctBase.h](#)

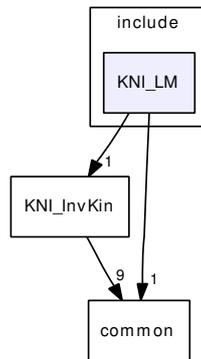
9.4 include/KNI_InvKin/ Directory Reference



Files

- file [ikBase.h](#)
- file [KatanaKinematics.h](#)
- file [KatanaKinematics5M180.h](#)
- file [KatanaKinematics6M180.h](#)
- file [KatanaKinematics6M90G.h](#)
- file [KatanaKinematics6M90T.h](#)
- file [KatanaKinematicsDecisionAlgorithms.h](#)

9.5 include/KNI_LM/ Directory Reference



Files

- file [lmBase.h](#)

Chapter 10

KatanaNativeInterface Namespace Documentation

10.1 KNI Namespace Reference

Classes

- class [Timer](#)
Provides a stop-watch-like class with a resolution of milliseconds.
- class [kmlFactory](#)
This class is for internal use only It may change at any time It shields the configuration file parsing.
- class [NoSolutionException](#)
No solution found for the given cartesian coordinates.
- struct [KinematicParameters](#)
To pass different parameters for the kinematic implementations.
- class [KatanaKinematics](#)
The base class for all kinematic implementations.
- class [KatanaKinematics5M180](#)
Author:
Tiziano Mueller <tiziano.mueller@neuronics.ch>
- class [KatanaKinematics6M180](#)
Author:
Tiziano Mueller <tiziano.mueller@neuronics.ch>
- class [KatanaKinematics6M90G](#)
Author:
Tiziano Mueller <tiziano.mueller@neuronics.ch>
- class [KatanaKinematics6M90T](#)

Author:

Tiziano Mueller <tiziano.mueller@neuronics.ch>

- struct [KinematicsDefaultEncMinAlgorithm](#)

Functions

- void [sleep](#) (long time)

This functions shields the platform specific implementation of the sleep function.

10.1.1 Function Documentation**10.1.1.1 void KNI::sleep (long *time*)**

This functions shields the platform specific implementation of the sleep function.

10.2 KNI_MHF Namespace Reference

Classes

- struct [unary_precalc_sin](#)
function-object which calculates sinus for n-elements of a container if used together with a STL algorithm
- struct [unary_precalc_cos](#)
See also:
[unary_precalc_sin](#)
- struct [unary_rad2deg](#)
a function-object version of rad2deg
- struct [unary_deg2rad](#)
a function-object version of rad2deg

Functions

- template<typename `_T`> short [sign](#) (`_T` x)
- template<typename `_T`> `_T` [atan1](#) (`_T` in1, `_T` in2)
- template<typename `_T`> `_T` [acotan](#) (const `_T` in)
- template<typename `_T`> `_T` [atan0](#) (const `_T` in1, const `_T` in2)
- template<typename `_T`> `_T` [pow2](#) (const `_T` in)
- template<typename `_T`> `_T` [rad2deg](#) (const `_T` a)
conversion from radian to degree
- template<typename `_T`> `_T` [deg2rad](#) (const `_T` a)
conversion from degree to radian
- template<typename `_T`> `_T` [anglereduce](#) (const `_T` a)
- template<typename `_angleT`, typename `_encT`> `_encT` [rad2enc](#) (`_angleT` const &angle, `_angleT` const &angleOffset, `_encT` const &enc, `_encT` const &encOffset, `_encT` const &rotDir)
converts absolute angles in radian to encoders.
- template<typename `_angleT`, typename `_encT`> `_angleT` [enc2rad](#) (`_encT` const &enc, `_angleT` const &angleOffset, `_encT` const &enc, `_encT` const &encOffset, `_encT` const &rotDir)
converts encoders to absolute angles in radian
- double [findFirstEqualAngle](#) (double cosValue, double sinValue, double tolerance)
Find the first equal angle.

10.2.1 Function Documentation

10.2.1.1 template<typename `_T`> `_T` KNI_MHF::acotan (const `_T` in) [inline]

Definition at line 77 of file MathHelperFunctions.h.

References `M_PI`.

10.2.1.2 `template<typename _T> _T KNI_MHF::anglereduce (const _T a) [inline]`

Definition at line 126 of file MathHelperFunctions.h.

References `M_PI`.

Referenced by `findFirstEqualAngle()`.

10.2.1.3 `template<typename _T> _T KNI_MHF::atan0 (const _T in1, const _T in2) [inline]`

Definition at line 85 of file MathHelperFunctions.h.

References `M_PI`.

10.2.1.4 `template<typename _T> _T KNI_MHF::atan1 (_T in1, _T in2) [inline]`

Definition at line 62 of file MathHelperFunctions.h.

References `M_PI`, and `sign()`.

Here is the call graph for this function:



10.2.1.5 `template<typename _T> _T KNI_MHF::deg2rad (const _T a) [inline]`

conversion from degree to radian

Definition at line 114 of file MathHelperFunctions.h.

References `M_PI`.

Referenced by `KNI_MHF::unary_deg2rad<_T>::operator()`.

10.2.1.6 `template<typename _angleT, typename _encT> _angleT KNI_MHF::enc2rad (_encT const & enc, _angleT const & angleOffset, _encT const & enc, _encT const & encOffset, _encT const & rotDir) [inline]`

converts encoders to absolute angles in radian

Definition at line 148 of file MathHelperFunctions.h.

References `M_PI`.

10.2.1.7 `double KNI_MHF::findFirstEqualAngle (double cosValue, double sinValue, double tolerance) [inline]`

Find the first equal angle.

You have to pass a cos and a sin Value

Definition at line 157 of file MathHelperFunctions.h.

References `anglereduce()`, and `M_PI`.

Here is the call graph for this function:



10.2.1.8 `template<typename _T> _T KNI_MHF::pow2 (const _T in) [inline]`

Definition at line 92 of file MathHelperFunctions.h.

10.2.1.9 `template<typename _T> _T KNI_MHF::rad2deg (const _T a) [inline]`

conversion from radian to degree

Definition at line 100 of file MathHelperFunctions.h.

References `M_PI`.

Referenced by `KNI_MHF::unary_rad2deg<_T >::operator()()`.

10.2.1.10 `template<typename _angleT, typename _encT> _encT KNI_MHF::rad2enc (_angleT const & angle, _angleT const & angleOffset, _encT const & epc, _encT const & encOffset, _encT const & rotDir) [inline]`

converts absolute angles in radian to encoders.

Definition at line 134 of file MathHelperFunctions.h.

References `M_PI`.

10.2.1.11 `template<typename _T> short KNI_MHF::sign (_T x) [inline]`

Definition at line 37 of file MathHelperFunctions.h.

Referenced by `atan1()`.

Chapter 11

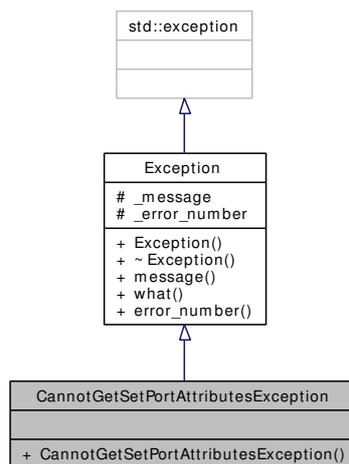
KatanaNativeInterface Class Documentation

11.1 CannotGetSetPortAttributesException Class Reference

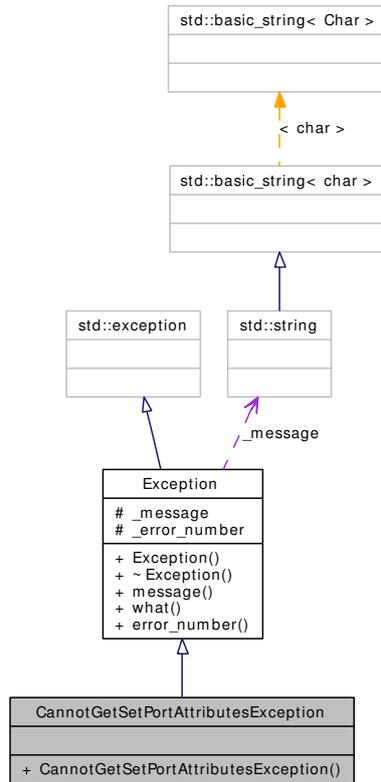
Could not set or get the attributes for the given serial communication device.

```
#include <cdlCOMExceptions.h>
```

Inheritance diagram for CannotGetSetPortAttributesException:



Collaboration diagram for CannotGetSetPortAttributesException:



Public Member Functions

- [CannotGetSetPortAttributesException](#) (const std::string &port) throw ()

11.1.1 Detailed Description

Could not set or get the attributes for the given serial communication device.

Note:

error_number=-11

Definition at line 56 of file cdICOMExceptions.h.

11.1.2 Constructor & Destructor Documentation

11.1.2.1 [CannotGetSetPortAttributesException::CannotGetSetPortAttributesException](#) (const std::string &port) throw () [inline]

Definition at line 58 of file cdICOMExceptions.h.

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

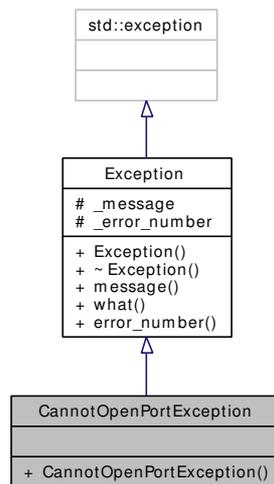
- include/KNI/[cdICOMExceptions.h](#)

11.2 CannotOpenPortException Class Reference

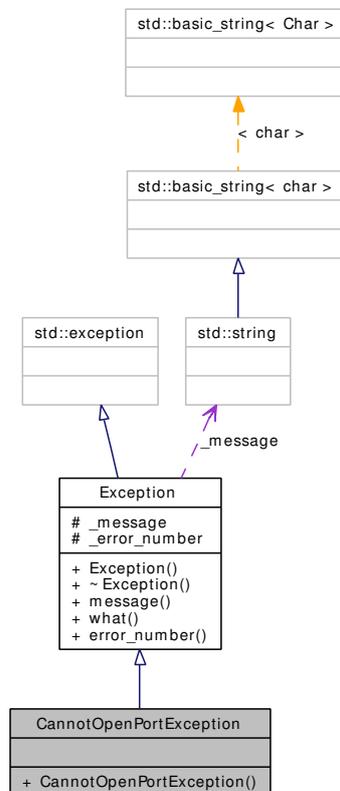
Failed to open the serial communication device.

```
#include <cdlCOMExceptions.h>
```

Inheritance diagram for CannotOpenPortException:



Collaboration diagram for CannotOpenPortException:



Public Member Functions

- [CannotOpenPortException](#) (const std::string &port, const std::string os_msg) throw ()

11.2.1 Detailed Description

Failed to open the serial communication device.

Note:

error_number=-10

Linux only: You get also the direct error message from the system

Definition at line 47 of file cdlCOMExceptions.h.

11.2.2 Constructor & Destructor Documentation

11.2.2.1 CannotOpenPortException::CannotOpenPortException (const std::string &port, const std::string os_msg) throw () [inline]

Definition at line 49 of file cdlCOMExceptions.h.

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

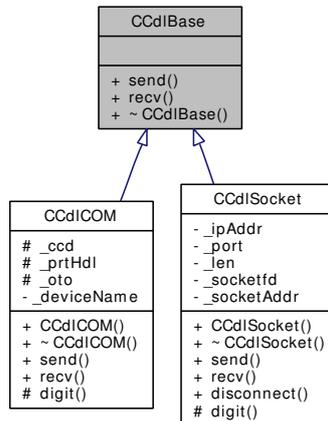
- [include/KNI/cdlCOMExceptions.h](#)

11.3 CCdlBase Class Reference

Abstract base class for devices.

```
#include <cdlBase.h>
```

Inheritance diagram for CCdlBase:



Public Member Functions

- virtual int `send` (const void *_buf, int _sz)=0
Pure function to send data.
- virtual int `recv` (void *_buf, int _sz)=0
Pure function to receive data.
- virtual `~CCdlBase` ()
destructor

11.3.1 Detailed Description

Abstract base class for devices.

This class is the base abstract class for devices; the abbreviation 'cdl' stands for 'Communication Device Layer'. By inheriting from this class different communication devices such a USB or a COM port can be handled easier.

Definition at line 47 of file cdlBase.h.

11.3.2 Constructor & Destructor Documentation

11.3.2.1 virtual CCdlBase::~~CCdlBase () [inline, virtual]

destructor

This class is only an interface

Definition at line 69 of file `cdlBase.h`.

11.3.3 Member Function Documentation

11.3.3.1 `virtual int CCdlBase::send (const void * _buf, int _sz)` [pure virtual]

Pure function to send data.

This function is pure and should always be overwritten by classes inheriting from 'CCdlBase'. As the name proposes the function should contain a sending behaviour from the device.

Implemented in [CCdlCOM](#), and [CCdlSocket](#).

11.3.3.2 `virtual int CCdlBase::recv (void * _buf, int _sz)` [pure virtual]

Pure function to receive data.

This function is pure and should always be overwritten by classes inheriting from 'CCdlBase'. As the name proposes the function should contain a sending behaviour from the device.

Implemented in [CCdlCOM](#), and [CCdlSocket](#).

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

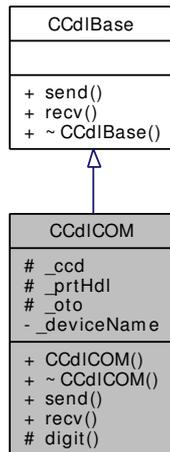
- [include/KNI/cdlBase.h](#)

11.4 CCdICOM Class Reference

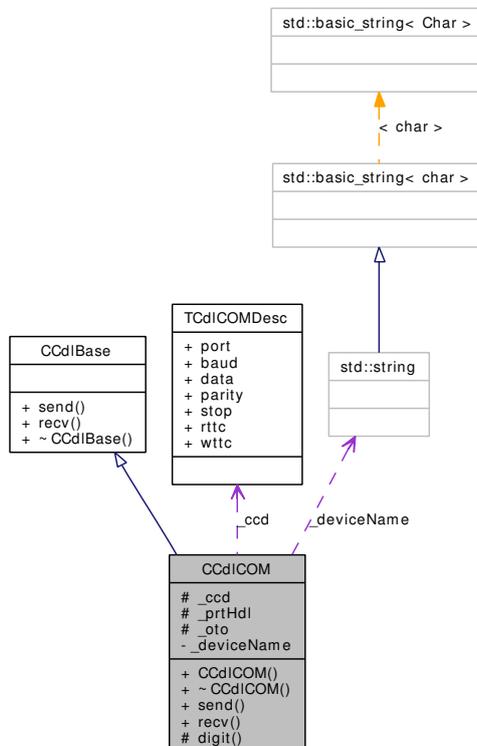
Encapsulates the serial port device.

```
#include <cdlCOM.h>
```

Inheritance diagram for CCdICOM:



Collaboration diagram for CCdICOM:



Public Member Functions

- [CCdICOM](#) ([TCdICOMDesc](#) ccd)
Construct a [CCdICOM](#) class.
- virtual [~CCdICOM](#) ()
Destructs the class.
- virtual int [send](#) (const void *buf, int size)
Sends data to the device.
- virtual int [recv](#) (void *buf, int size)
Receives data from the device.

Static Protected Member Functions

- static char [digit](#) (const int _val)
Converts an integer to a char.

Protected Attributes

- [TCdICOMDesc _ccd](#)
Stores the attributes of the serial port device.
- int [_prtHdl](#)
port handle
- [termios _oto](#)
old timeouts

Private Attributes

- [std::string _deviceName](#)

11.4.1 Detailed Description

Encapsulates the serial port device.

This class is responsible for direct communication with the serial port device. It builds the lowest layer for communication and uses the system API functions to get access the to the device.

Definition at line 73 of file [cdICOM.h](#).

11.4.2 Constructor & Destructor Documentation

11.4.2.1 CCdlCOM::CCdlCOM (TCdlCOMDesc *ccd*)

Construct a [CCdlCOM](#) class.

To this constructor a 'TCdlCOMDesc' parameter has to be given, which describes the desired serial port. An attempt to open a connection to the desired device will be tried.

11.4.2.2 virtual CCdlCOM::~~CCdlCOM () [virtual]

Destructs the class.

11.4.3 Member Function Documentation

11.4.3.1 static char CCdlCOM::digit (const int *_val*) [inline, static, protected]

Converts an integer to a char.

Definition at line 99 of file `cdlCOM.h`.

11.4.3.2 virtual int CCdlCOM::send (const void * *buf*, int *size*) [virtual]

Sends data to the device.

Implements [CCdlBase](#).

11.4.3.3 virtual int CCdlCOM::recv (void * *buf*, int *size*) [virtual]

Receives data from the device.

Implements [CCdlBase](#).

11.4.4 Member Data Documentation

11.4.4.1 std::string CCdlCOM::_deviceName [private]

Definition at line 75 of file `cdlCOM.h`.

11.4.4.2 TCdlCOMDesc CCdlCOM::_ccd [protected]

Stores the attributes of the serial port device.

Definition at line 79 of file `cdlCOM.h`.

11.4.4.3 int CCdlCOM::_prtHdl [protected]

port handle

Definition at line 89 of file `cdlCOM.h`.

11.4.4.4 struct termios CCdlCOM::_oto [protected]

old timeouts

Definition at line 90 of file cdlCOM.h.

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

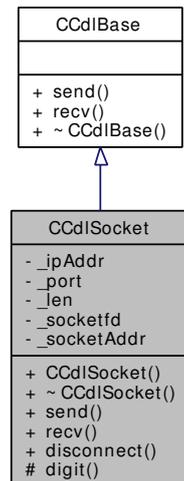
- [include/KNI/cdlCOM.h](#)

11.5 CCdlSocket Class Reference

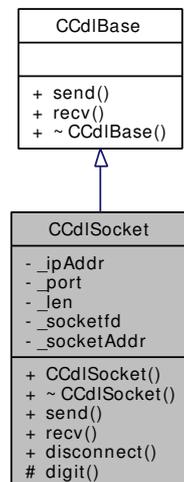
Encapsulates the socket communication device.

```
#include <cdlSocket.h>
```

Inheritance diagram for CCdlSocket:



Collaboration diagram for CCdlSocket:



Public Member Functions

- [CCdlSocket](#) (char *address, int port)
Constructs a [CCdlSocket](#) object.
- virtual [~CCdlSocket](#) ()
Destructs the object.

- virtual int [send](#) (const void *_buf, int _size)
Sends data to the socket.
- virtual int [recv](#) (void *_buf, int _size)
Receives data from the socket.
- virtual int [disconnect](#) ()
Terminates the socket connection.

Static Protected Member Functions

- static char [digit](#) (const int _val)
Converts an integer to a char.

Private Attributes

- char * [_ipAddr](#)
IP Address of the Robot or simulation environment.
- int [_port](#)
Port number of the [KNI](#) communication socket.
- int [_len](#)
Length of the message.
- int [_socketfd](#)
File handler for the socket.
- sockaddr_in [_socketAddr](#)
Structure to fill in the socket communication parameters.

11.5.1 Detailed Description

Encapsulates the socket communication device.

This class is responsible for direct communication with the Katana robot or its simulation environment through sockets. It builds the lowest layer for [KNI](#) communication and uses the system API functions to get access to the socket.

Definition at line 61 of file `cdlSocket.h`.

11.5.2 Constructor & Destructor Documentation

11.5.2.1 `CCdlSocket::CCdlSocket (char * adress, int port)`

Constructs a [CCdlSocket](#) object.

To this constructor the socket's AF_INET address (for platform independence) and port number have to be given as parameters. An attempt to open a connection to the desired device will be tried and if successful, 'lastOP()' will return 'lopDONE', otherwise 'lopFAIL'.

11.5.2.2 virtual CCdlSocket::~~CCdlSocket () [virtual]

Destructs the object.

11.5.3 Member Function Documentation

11.5.3.1 static char CCdlSocket::digit (const int *val*) [inline, static, protected]

Converts an integer to a char.

Definition at line 93 of file cdlSocket.h.

11.5.3.2 virtual int CCdlSocket::send (const void * *buf*, int *size*) [virtual]

Sends data to the socket.

Implements [CCdlBase](#).

11.5.3.3 virtual int CCdlSocket::recv (void * *buf*, int *size*) [virtual]

Receives data from the socket.

Implements [CCdlBase](#).

11.5.3.4 virtual int CCdlSocket::disconnect () [virtual]

Terminates the socket connection.

11.5.4 Member Data Documentation

11.5.4.1 char* CCdlSocket::_ipAddr [private]

IP Address of the Robot or simulation environment.

Set to localhost or 127.0.0.1 if the simulation runs on the same machine

Definition at line 65 of file cdlSocket.h.

11.5.4.2 int CCdlSocket::_port [private]

Port number of the [KNI](#) communication socket.

Definition at line 67 of file cdlSocket.h.

11.5.4.3 int CCdlSocket::_len [private]

Length of the message.

Definition at line 69 of file cdlSocket.h.

11.5.4.4 int CCdlSocket::_socketfd [private]

File handler for the socket.

Definition at line 82 of file cdlSocket.h.

11.5.4.5 struct sockaddr_in CCdlSocket::_socketAddr [private]

Structure to fill in the socket communication parameters.

Definition at line 84 of file cdlSocket.h.

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

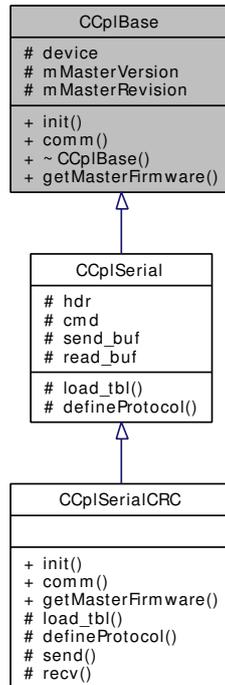
- [include/KNI/cdlSocket.h](#)

11.6 CCplBase Class Reference

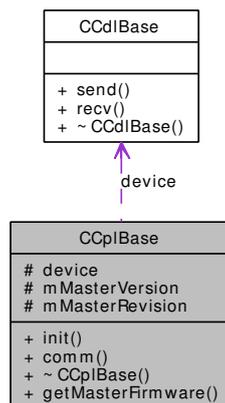
Abstract base class for protocol definiton.

```
#include <cplBase.h>
```

Inheritance diagram for CCplBase:



Collaboration diagram for CCplBase:



Public Member Functions

- virtual bool `init(CCdlBase *_device, byte _kataddr=24)=0`
Basic initializing function.

- virtual void `comm` (const `byte` *pack, `byte` *buf, `byte` *size)=0
Base communication function.
- virtual `~CCplBase` ()
destructor
- virtual void `getMasterFirmware` (short *fw, short *rev)=0
Get the master firmware of the robot we are communicating with.

Protected Attributes

- `CCdlBase` * `device`
communication device
- short `mMasterVersion`
master version of robot we are communicating with
- short `mMasterRevision`
master firmware revision

11.6.1 Detailed Description

Abstract base class for protocol definiton.

The robot can be controlled by using different kind of protocols; this class has been introduced as an abstract base class to manage them gether; every protocol the robot should use in futur should be derived from this class.

Definition at line 47 of file `cplBase.h`.

11.6.2 Constructor & Destructor Documentation

11.6.2.1 virtual `CCplBase::~CCplBase` () [inline, virtual]

destructor

This class is only an interface

Definition at line 75 of file `cplBase.h`.

11.6.3 Member Function Documentation

11.6.3.1 virtual `bool CCplBase::init` (`CCdlBase` * `_device`, `byte` `_kataddr` = 24) [pure virtual]

Basic initializing function.

The children of this class should write their initializing part in that function.

Implemented in `CCplSerialCRC`.

11.6.3.2 virtual void CCplBase::comm (const **byte** * *pack*, **byte** * *buf*, **byte** * *size*) [pure virtual]

Base communication function.

The children of this class should write their main double way communication in this function.

Implemented in [CCplSerialCRC](#).

11.6.3.3 virtual void CCplBase::getMasterFirmware (short * *fw*, short * *rev*) [pure virtual]

Get the master firmware of the robot we are communicating with.

Get master firmware read at initialization time.

Implemented in [CCplSerialCRC](#).

11.6.4 Member Data Documentation

11.6.4.1 CCdlBase* CCplBase::device [protected]

communication device

Definition at line 50 of file [cplBase.h](#).

11.6.4.2 short CCplBase::mMasterVersion [protected]

master version of robot we are communicating with

Definition at line 51 of file [cplBase.h](#).

11.6.4.3 short CCplBase::mMasterRevision [protected]

master firmware revision

Definition at line 52 of file [cplBase.h](#).

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

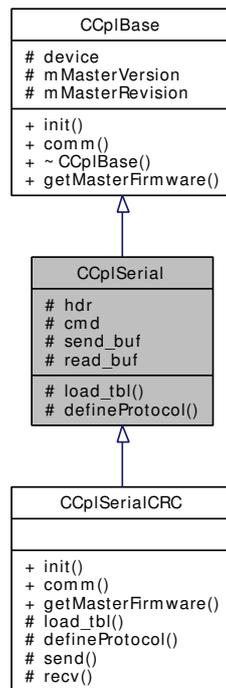
- [include/KNI/cplBase.h](#)

11.7 CCplSerial Class Reference

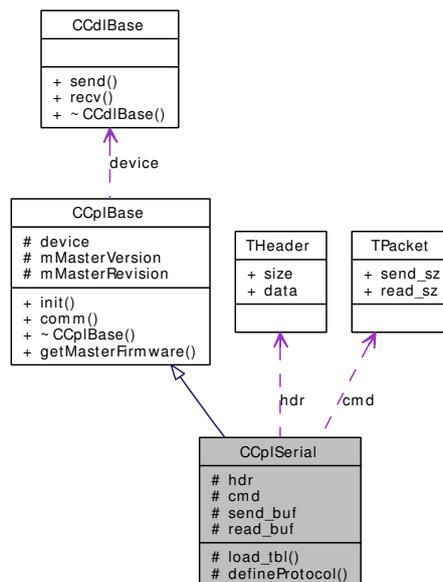
Base class of two different serial protocols.

```
#include <cplSerial.h>
```

Inheritance diagram for CCplSerial:



Collaboration diagram for CCplSerial:



Protected Member Functions

- virtual bool `load_tbl` ()=0
Loads the command table from the robot's firmware.
- virtual void `defineProtocol` (byte _kataddr)=0
Defines the protocol's attributes.

Protected Attributes

- `THeader hdr`
header
- `TPacket cmd` [256]
command table
- `byte send_buf` [256]
sending buffer
- `byte read_buf` [256]
receive buffer

11.7.1 Detailed Description

Base class of two different serial protocols.

Definition at line 73 of file `cplSerial.h`.

11.7.2 Member Function Documentation

11.7.2.1 virtual bool `CCplSerial::load_tbl` () [protected, pure virtual]

Loads the command table from the robot's firmware.

Implemented in `CCplSerialCRC`.

11.7.2.2 virtual void `CCplSerial::defineProtocol` (byte _kataddr) [protected, pure virtual]

Defines the protocol's attributes.

Implemented in `CCplSerialCRC`.

11.7.3 Member Data Documentation

11.7.3.1 `THeader CCplSerial::hdr` [protected]

header

Definition at line 76 of file `cplSerial.h`.

11.7.3.2 TPacket CCplSerial::cmd[256] [protected]

command table

Definition at line 77 of file cplSerial.h.

11.7.3.3 byte CCplSerial::send_buf[256] [protected]

sending buffer

Definition at line 79 of file cplSerial.h.

11.7.3.4 byte CCplSerial::read_buf[256] [protected]

receive buffer

Definition at line 80 of file cplSerial.h.

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

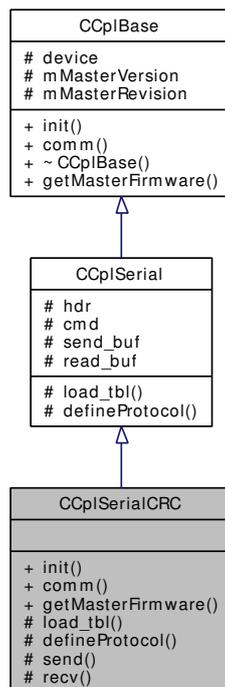
- [include/KNI/cplSerial.h](#)

11.8 CCplSerialCRC Class Reference

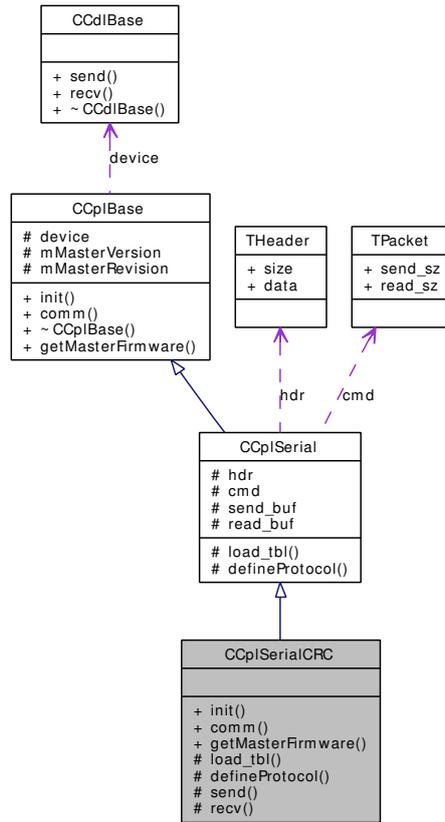
Implement the Serial-Zero protocol Initializing functionCommunication functionImplement the Serial-CRC protocol.

```
#include <cplSerial.h>
```

Inheritance diagram for CCplSerialCRC:



Collaboration diagram for CCplSerialCRC:



Public Member Functions

- virtual bool `init` (`CCdlBase *_device`, `byte _kataddr=24`)
Initializing function.
- virtual void `comm` (`const byte *pack`, `byte *buf`, `byte *size`)
Communication function.
- virtual void `getMasterFirmware` (`short *fw`, `short *rev`)
Get the master firmware of the robot we are communicating with.

Protected Member Functions

- virtual bool `load_tbl` ()
Loads the command table from the robot's firmware.
- virtual void `defineProtocol` (`byte _kataddr`)
Defines the protocol's attributes.
- virtual void `send` (`byte *send_buf`, `byte write_sz`, `short retries=3`)
- virtual void `recv` (`byte *read_buf`, `byte read_sz`, `byte *size`)

11.8.1 Detailed Description

Implement the Serial-Zero protocol Initializing functionCommunication functionImplement the Serial-CRC protocol.

Definition at line 118 of file cplSerial.h.

11.8.2 Member Function Documentation

11.8.2.1 `virtual bool CCplSerialCRC::load_tbl ()` [protected, virtual]

Loads the command table from the robot's firmware.

Implements [CCplSerial](#).

11.8.2.2 `virtual void CCplSerialCRC::defineProtocol (byte _kataddr)` [protected, virtual]

Defines the protocol's attributes.

Implements [CCplSerial](#).

11.8.2.3 `virtual void CCplSerialCRC::send (byte * send_buf, byte write_sz, short retries = 3)` [protected, virtual]

11.8.2.4 `virtual void CCplSerialCRC::recv (byte * read_buf, byte read_sz, byte * size)` [protected, virtual]

11.8.2.5 `virtual bool CCplSerialCRC::init (CCdlBase * _device, byte _kataddr = 24)` [virtual]

Initializing function.

Init the protocols basic attributes.

Implements [CCplBase](#).

11.8.2.6 `virtual void CCplSerialCRC::comm (const byte * pack, byte * buf, byte * size)` [virtual]

Communication function.

Sends a communications packet and receives one from the robot.

Implements [CCplBase](#).

11.8.2.7 `virtual void CCplSerialCRC::getMasterFirmware (short * fw, short * rev)` [virtual]

Get the master firmware of the robot we are communicating with.

Get master firmware read at initialization time.

Implements [CCplBase](#).

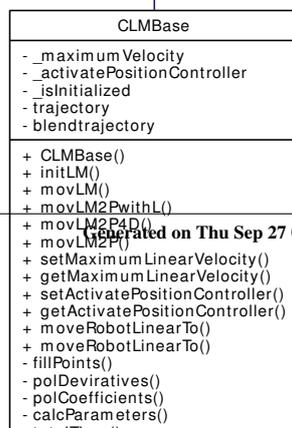
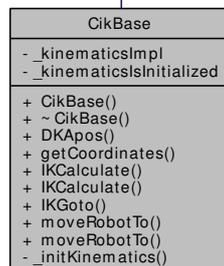
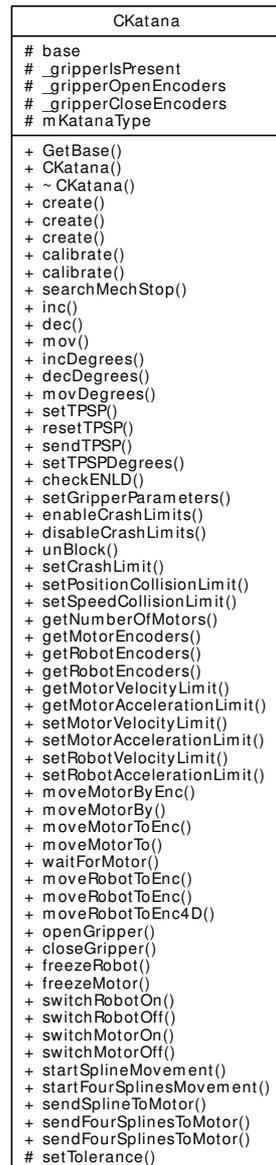
The documentation for this class was generated from the following file:

- [include/KNI/cplSerial.h](#)

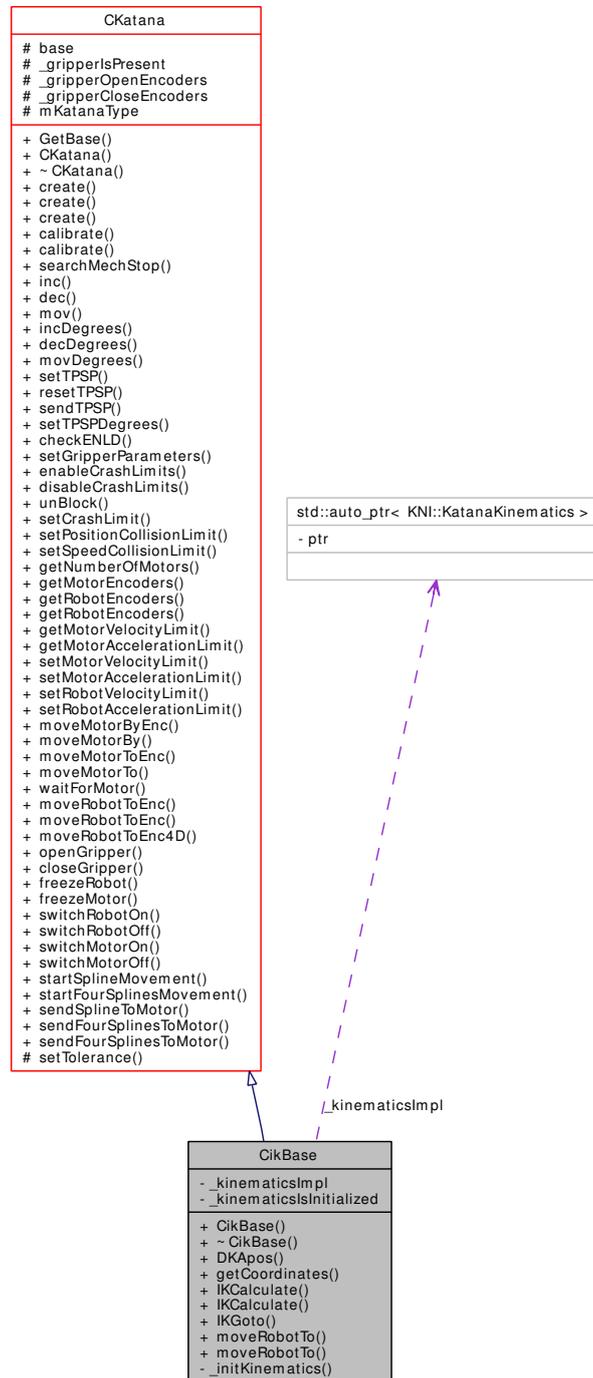
11.9 CikBase Class Reference

```
#include <ikBase.h>
```

Inheritance diagram for CikBase:



Collaboration diagram for CikBase:



Public Member Functions

- `CikBase ()`
- `~CikBase ()`
- `void DKAprs (double *position)`

Returns the current position of the robot in cartesian coordinates.

- void [getCoordinates](#) (double &x, double &y, double &z, double &phi, double &theta, double &psi, bool refreshEncoders=true)

Returns the current position of the robot in cartesian coordinates.

- void [IKCalculate](#) (double X, double Y, double Z, double A1, double Be, double Ga, std::vector< int >::iterator solution_iter)

Calculates a set of encoders for the given coordinates.

- void [IKCalculate](#) (double X, double Y, double Z, double A1, double Be, double Ga, std::vector< int >::iterator solution_iter, const std::vector< int > &actualPosition)

Calculates a set of encoders for the given coordinates.

- void [IKGoto](#) (double X, double Y, double Z, double A1, double Be, double Ga, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)

Moves to robot to given cartesian coordinates and euler-angles.

- void [moveRobotTo](#) (double x, double y, double z, double phi, double theta, double psi, bool waitUntilReached=false, int waitTimeout=TM_ENDLESS)

Moves to robot to given cartesian coordinates and euler-angles.

- void [moveRobotTo](#) (std::vector< double > coordinates, bool waitUntilReached=false, int waitTimeout=TM_ENDLESS)

This method does the same as the one above and is mainly provided for convenience.

Private Member Functions

- void [_initKinematics](#) ()

Private Attributes

- std::auto_ptr< [KNI::KatanaKinematics](#) > [_kinematicsImpl](#)
- bool [_kinematicsIsInitialized](#)

11.9.1 Detailed Description

Definition at line 44 of file ikBase.h.

11.9.2 Constructor & Destructor Documentation

11.9.2.1 [CikBase::CikBase](#) () [inline]

Definition at line 53 of file ikBase.h.

11.9.2.2 [CikBase::~~CikBase](#) () [inline]

Definition at line 54 of file ikBase.h.

11.9.3 Member Function Documentation

11.9.3.1 void CikBase::_initKinematics () [private]

11.9.3.2 void CikBase::DKApos (double * *position*)

Returns the current position of the robot in cartesian coordinates.

Note:

This method is deprecated, please use getCoordinates(...) instead

11.9.3.3 void CikBase::getCoordinates (double & *x*, double & *y*, double & *z*, double & *phi*, double & *theta*, double & *psi*, bool *refreshEncoders* = true)

Returns the current position of the robot in cartesian coordinates.

Parameters:

refreshEncoders With this parameter you can determine if the method reads the actual encoders from the robot or if it will use the cached ones

Note:

This function returns a tuple in python

11.9.3.4 void CikBase::IKCalculate (double *X*, double *Y*, double *Z*, double *Al*, double *Be*, double *Ga*, std::vector< int >::iterator *solution_iter*)

Calculates a set of encoders for the given coordinates.

This method reads the current encoders from the robot and involves therefore also communication to the robot

11.9.3.5 void CikBase::IKCalculate (double *X*, double *Y*, double *Z*, double *Al*, double *Be*, double *Ga*, std::vector< int >::iterator *solution_iter*, const std::vector< int > & *actualPosition*)

Calculates a set of encoders for the given coordinates.

For this method you have to pass an actualPosition too. No communication with the robot will be done here.

11.9.3.6 void CikBase::IKGoto (double *X*, double *Y*, double *Z*, double *Al*, double *Be*, double *Ga*, bool *wait* = false, int *tolerance* = 100, long *timeout* = TM_ENDLESS)

Moves to robot to given cartesian coordinates and euler-angles.

Note:

This method is deprecated, please use moveRobotTo(...) instead

11.9.3.7 `void CikBase::moveRobotTo (double x, double y, double z, double phi, double theta, double psi, bool waitUntilReached = false, int waitTimeout = TM_ENDLESS)`

Moves to robot to given cartesian coordinates and euler-angles.

Note:

Instead of a given tolerance, a default tolerance is being used

11.9.3.8 `void CikBase::moveRobotTo (std::vector< double > coordinates, bool waitUntilReached = false, int waitTimeout = TM_ENDLESS)`

This method does the same as the one above and is mainly provided for convenience.

Note:

You can call this function in python using tuples: Example: `katana.moveRobotTo((x,y,z,phi,theta,psi))`

If the size of the container is smaller than 6, it will throw an exception

11.9.4 Member Data Documentation

11.9.4.1 `std::auto_ptr<KNI::KatanaKinematics> CikBase::_kinematicsImpl` [private]

Definition at line 47 of file ikBase.h.

11.9.4.2 `bool CikBase::_kinematicsIsInitialized` [private]

Definition at line 48 of file ikBase.h.

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

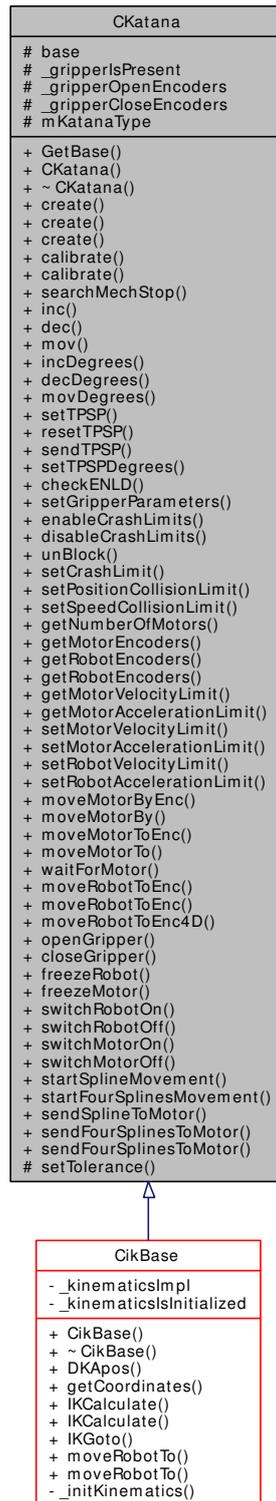
- [include/KNI_InvKin/ikBase.h](#)

11.10 CKatana Class Reference

Extended Katana class with additional functions.

```
#include <kmlExt.h>
```

Inheritance diagram for CKatana:



Collaboration diagram for CKatana:

```

class CKatBase
{
public:
    # gnl
    # mfw
    # ids
    # ctb
    # cbx
    # ech
    # mot
    # sct
    # eff
    # protocol
    # mMasterVersion
    # mMasterRevision

    + GetGNL()
    + GetMFW()
    + GetIDS()
    + GetCTB()
    + GetCBX()
    + GetECH()
    + GetMOT()
    + GetSCT()
    + GetEFF()
    + CKatBase()
    + ~CKatBase()
    + init()
    + recvMFW()
    + recvIDS()
    + recvCTB()
    + recvGMS()
    + recvCBX()
    + recvECH()
    + recvNMP()
    + recvMPS()
    + getProtocol()
    + checkKatanaType()
    + sendCBX()
    + sendTPSP()
    + getMasterFirmware()
    + enableCrashLimits()
    + disableCrashLimits()
    + unBlock()
    + setCrashLimit()
    + setPositionCollisionLimit()
    + setSpeedCollisionLimit()
    + startSplineMovement()
    + startFourSplinesMovement()
    + sendSLMP()
    + sendSLM()
}
    
```

```

class CKatana
{
public:
    # base
    # _gripperIsPresent
    # _gripperOpenEncoders
    # _gripperCloseEncoders
    # mKatanaType

    + GetBase()
    + CKatana()
    + ~CKatana()
    + create()
    + create()
    + create()
    + calibrate()
    + calibrate()
    + searchMechStop()
    + inc()
    + dec()
    + mov()
    + incDegrees()
    + decDegrees()
    + movDegrees()
    + setTPSP()
    + resetTPSP()
    + sendTPSP()
    + setTPSPDegrees()
    + checkENLD()
    + setGripperParameters()
    + enableCrashLimits()
    + disableCrashLimits()
    + unBlock()
    + setCrashLimit()
    + setPositionCollisionLimit()
    + setSpeedCollisionLimit()
    + getNumberOfMotors()
    + getMotorEncoders()
    + getRobotEncoders()
    + getRobotEncoders()
    + getMotorVelocityLimit()
    + getMotorAccelerationLimit()
    + setMotorVelocityLimit()
    + setMotorAccelerationLimit()
    + setRobotVelocityLimit()
    + setRobotAccelerationLimit()
    + moveMotorByEnc()
    + moveMotorToEnc()
    + moveMotorTo()
    + waitForMotor()
    + moveRobotToEnc()
    + moveRobotToEnc()
    + moveRobotToEnc4D()
    + openGripper()
    + closeGripper()
    + freezeRobot()
    + freezeMotor()
    + switchRobotOn()
}
    
```



Public Member Functions

- [CKatBase * GetBase \(\)](#)
Returns pointer to 'CKatBase'.*
- [CKatana \(\)](#)
Constructor.
- [~CKatana \(\)](#)
Destructor.
- void [create](#) (const char *configurationFile, [CCplBase](#) *protocol)
Create routine.
- void [create](#) ([KNI::kmlFactory](#) *infos, [CCplBase](#) *protocol)
- void [create](#) ([TKatGNL](#) &gnl, [TKatMOT](#) &mot, [TKatSCT](#) &sct, [TKatEFF](#) &eff, [CCplBase](#) *protocol)
Create routine.
- void [calibrate](#) ()
- void [calibrate](#) (long idx, [TMotCLB](#) clb, [TMotSCP](#) scp, [TMotDYL](#) dyl)
- void [searchMechStop](#) (long idx, [TSearchDir](#) dir, [TMotSCP](#) scp, [TMotDYL](#) dyl)
- void [inc](#) (long idx, int dif, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Increments the motor specified by an index postion in encoders.
- void [dec](#) (long idx, int dif, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Decrements the motor specified by an index postion in encoders.
- void [mov](#) (long idx, int tar, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Moves the motor specified by an index to a given target position in encoders.
- void [incDegrees](#) (long idx, double dif, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Increments the motor specified by an index postion in degree units.
- void [decDegrees](#) (long idx, double dif, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Decrements the motor specified by an index postion in degree units.
- void [movDegrees](#) (long idx, double tar, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Moves the motor specified by an index to a given target position in degree units.
- void [setTPSP](#) (long idx, int tar)
Sets the target position of a motor in encoders and allows the movement of that motor during the parallel movement.
- void [resetTPSP](#) ()
Forbid the movement of all the motors during the parallel movement.
- void [sendTPSP](#) (bool wait=false, long timeout=TM_ENDLESS)

Moves the allowed motors simultaneously.

- void [setTPSPDegrees](#) (long idx, double tar)
Sets the target position of a motor in degree Units and allows the movement of that motor during the parallel movement.
- bool [checkENLD](#) (long idx, double degrees)
Check if the absolute position in degrees is out of range.
- void [setGripperParameters](#) (bool isPresent, int openEncoders, int closeEncoders)
Tell the robot about the presence of a gripper.
- void [enableCrashLimits](#) ()
crash limits enable
- void [disableCrashLimits](#) ()
crash limits disable
- void [unBlock](#) ()
unblock robot after a crash
- void [setCrashLimit](#) (long idx, int limit)
unblock robot after a crash
- void [setPositionCollisionLimit](#) (long idx, int limit)
set collision position limits
- void [setSpeedCollisionLimit](#) (long idx, int limit)
set collision speed limits
- short [getNumberOfMotors](#) () const
- int [getMotorEncoders](#) (short number, bool refreshEncoders=true) const
- std::vector< int >::iterator [getRobotEncoders](#) (std::vector< int >::iterator start, std::vector< int >::const_iterator end, bool refreshEncoders=true) const
Write the cached encoders into the container.
- std::vector< int > [getRobotEncoders](#) (bool refreshEncoders=true) const
Get the current robot encoders as a vector-container.
- short [getMotorVelocityLimit](#) (short number) const
- short [getMotorAccelerationLimit](#) (short number) const
- void [setMotorVelocityLimit](#) (short number, short velocity)
- void [setMotorAccelerationLimit](#) (short number, short acceleration)
- void [setRobotVelocityLimit](#) (short velocity)
- void [setRobotAccelerationLimit](#) (short acceleration)
Set the velocity of all motors together.
- void [moveMotorByEnc](#) (short number, int encoders, bool waitUntilReached=false, int wait-Timeout=0)
- void [moveMotorBy](#) (short number, double radianAngle, bool waitUntilReached=false, int wait-Timeout=0)

- void [moveMotorToEnc](#) (short number, int encoders, bool waitUntilReached=false, int encTolerance=100, int waitTimeout=0)
- void [moveMotorTo](#) (short number, double radianAngle, bool waitUntilReached=false, int waitTimeout=0)
- void [waitForMotor](#) (short number, int encoders, int encTolerance=100, short mode=0, int waitTimeout=5000)
- void [moveRobotToEnc](#) (std::vector< int >::const_iterator start, std::vector< int >::const_iterator end, bool waitUntilReached=false, int encTolerance=100, int waitTimeout=0)

Move to robot to given encoders.

- void [moveRobotToEnc](#) (std::vector< int > encoders, bool waitUntilReached=false, int encTolerance=100, int waitTimeout=0)

Move to robot to given encoders in the vector-container.

- void [moveRobotToEnc4D](#) (std::vector< int > target, int velocity=180, int acceleration=1, int encTolerance=100)

Move to robot to given target in the vector-container with the given velocity, acceleration and tolerance.

- void [openGripper](#) (bool waitUntilReached=false, int waitTimeout=100)
- void [closeGripper](#) (bool waitUntilReached=false, int waitTimeout=100)
- void [freezeRobot](#) ()
- void [freezeMotor](#) (short number)
- void [switchRobotOn](#) ()
- void [switchRobotOff](#) ()
- void [switchMotorOn](#) (short number)
- void [switchMotorOff](#) (short number)
- void [startSplineMovement](#) (bool exactflag, int moreflag=1)

Start a spline movement.

- void [startFourSplinesMovement](#) (bool exactflag)

Start a fourSplines movement.

- void [sendSplineToMotor](#) (unsigned short number, short targetPosition, short duration, short p1, short p2, short p3, short p4)

Send one spline to the motor.

- void [sendFourSplinesToMotor](#) (unsigned short number, short targetPosition, short duration, std::vector< short > &coefficients)

Send four splines to the motor.

- void [sendFourSplinesToMotor](#) (unsigned short number, short targetPosition, short duration, short p01, short p11, short p21, short p31, short p02, short p12, short p22, short p32, short p03, short p13, short p23, short p33, short p04, short p14, short p24, short p34)

Protected Member Functions

- void [setTolerance](#) (long idx, int enc_tolerance)

Sets the tolerance range in encoder units for the robots movements.

Protected Attributes

- [CKatBase * base](#)

base katana

- [bool _gripperIsPresent](#)
- [int _gripperOpenEncoders](#)
- [int _gripperCloseEncoders](#)
- [int mKatanaType](#)

The type of KatanaXXX (300 or 400).

11.10.1 Detailed Description

Extended Katana class with additional functions.

This class uses the 'CKatBase* base' object to refer to a Katana robot.

Definition at line 64 of file kmlExt.h.

11.10.2 Constructor & Destructor Documentation

11.10.2.1 CKatana::CKatana () [inline]

Constructor.

Definition at line 86 of file kmlExt.h.

11.10.2.2 CKatana::~CKatana () [inline]

Destructor.

Definition at line 89 of file kmlExt.h.

11.10.3 Member Function Documentation

11.10.3.1 void CKatana::setTolerance (long idx, int enc_tolerance) [protected]

Sets the tolerance range in encoder units for the robots movements.

11.10.3.2 CKatBase* CKatana::GetBase () [inline]

Returns pointer to 'CKatBase*'.
Definition at line 81 of file kmlExt.h.

11.10.3.3 void CKatana::create (const char * configurationFile, CCplBase * protocol)

Create routine.

11.10.3.4 void CKatana::create (**KNI::kmlFactory** * *infos*, **CCplBase** * *protocol*)

11.10.3.5 void CKatana::create (**TKatGNL** & *gnl*, **TKatMOT** & *mot*, **TKatSCT** & *set*, **TKatEFF** & *eff*, **CCplBase** * *protocol*)

Create routine.

Parameters:

gnl katana initial attributes
mot motor initial attributes
set sensor controller initial attributes
eff end effector initial attributes
protocol protocol to be used

11.10.3.6 void CKatana::calibrate ()

11.10.3.7 void CKatana::calibrate (long *idx*, **TMotCLB** *clb*, **TMotSCP** *scp*, **TMotDYL** *dyl*)

Parameters:

idx motor index
clb calibration struct for one motor
scp static controller parameters
dyl dynamic controller parameters

11.10.3.8 void CKatana::searchMechStop (long *idx*, **TSearchDir** *dir*, **TMotSCP** *scp*, **TMotDYL** *dyl*)

Parameters:

idx motor index
dir search direction
scp static controller parameters
dyl dynamic controller parameters

11.10.3.9 void CKatana::inc (long *idx*, int *dif*, bool *wait* = false, int *tolerance* = 100, long *timeout* = TM_ENDLESS)

Increments the motor specified by an index position in encoders.

11.10.3.10 void CKatana::dec (long *idx*, int *dif*, bool *wait* = false, int *tolerance* = 100, long *timeout* = TM_ENDLESS)

Decrements the motor specified by an index position in encoders.

11.10.3.11 `void CKatana::mov (long idx, int tar, bool wait = false, int tolerance = 100, long timeout = TM_ENDLESS)`

Moves the motor specified by an index to a given target position in encoders.

11.10.3.12 `void CKatana::incDegrees (long idx, double dif, bool wait = false, int tolerance = 100, long timeout = TM_ENDLESS)`

Increments the motor specified by an index position in degree units.

11.10.3.13 `void CKatana::decDegrees (long idx, double dif, bool wait = false, int tolerance = 100, long timeout = TM_ENDLESS)`

Decrements the motor specified by an index position in degree units.

11.10.3.14 `void CKatana::movDegrees (long idx, double tar, bool wait = false, int tolerance = 100, long timeout = TM_ENDLESS)`

Moves the motor specified by an index to a given target position in degree units.

11.10.3.15 `void CKatana::setTPSP (long idx, int tar)`

Sets the target position of a motor in encoders and allows the movement of that motor during the parallel movement.

deprecated: for use with old Katana5M only

11.10.3.16 `void CKatana::resetTPSP ()`

Forbid the movement of all the motors during the parallel movement.

deprecated: for use with old Katana5M only

11.10.3.17 `void CKatana::sendTPSP (bool wait = false, long timeout = TM_ENDLESS)`

Moves the allowed motors simultaneously.

deprecated: for use with old Katana5M only

11.10.3.18 `void CKatana::setTPSPDegrees (long idx, double tar)`

Sets the target position of a motor in degree Units and allows the movement of that motor during the parallel movement.

deprecated: for use with old Katana5M only

11.10.3.19 `bool CKatana::checkENLD (long idx, double degrees)`

Check if the absolute position in degrees is out of range.

11.10.3.20 void CKatana::setGripperParameters (bool *isPresent*, int *openEncoders*, int *closeEncoders*)

Tell the robot about the presence of a gripper.

Parameters:

openEncoders Which encoders should be used as target positions for opening the gripper

closeEncoders Dito for closing the gripper

11.10.3.21 void CKatana::enableCrashLimits ()

crash limits enable

11.10.3.22 void CKatana::disableCrashLimits ()

crash limits disable

11.10.3.23 void CKatana::unBlock ()

unblock robot after a crash

11.10.3.24 void CKatana::setCrashLimit (long *idx*, int *limit*)

unblock robot after a crash

11.10.3.25 void CKatana::setPositionCollisionLimit (long *idx*, int *limit*)

set collision position limits

11.10.3.26 void CKatana::setSpeedCollisionLimit (long *idx*, int *limit*)

set collision speed limits

11.10.3.27 short CKatana::getNumberOfMotors () const

11.10.3.28 int CKatana::getMotorEncoders (short *number*, bool *refreshEncoders* = true) const

11.10.3.29 std::vector<int>::iterator CKatana::getRobotEncoders (std::vector< int >::iterator *start*, std::vector< int >::const_iterator *end*, bool *refreshEncoders* = true) const

Write the cached encoders into the container.

Set refreshEncoders=true if the [KNI](#) should fetch them from the robot. If m=distance(start, end) is smaller than the number of motors, only the first m motors will be written to the container, the function will not throw an exception because of this. The return value will point to one element after the last one.

11.10.3.30 `std::vector<int> CKatana::getRobotEncoders (bool refreshEncoders = true) const`

Get the current robot encoders as a vector-container.

This method is mainly provided for convenience. It is easier than the other `getRobotEncoders` method but probably not so efficient. It is much easier to use via the wrappers.

11.10.3.31 `short CKatana::getMotorVelocityLimit (short number) const`**11.10.3.32** `short CKatana::getMotorAccelerationLimit (short number) const`**11.10.3.33** `void CKatana::setMotorVelocityLimit (short number, short velocity)`**11.10.3.34** `void CKatana::setMotorAccelerationLimit (short number, short acceleration)`**11.10.3.35** `void CKatana::setRobotVelocityLimit (short velocity)`**11.10.3.36** `void CKatana::setRobotAccelerationLimit (short acceleration)`

Set the velocity of all motors together.

This does not set the velocity of the TCP.

11.10.3.37 `void CKatana::moveMotorByEnc (short number, int encoders, bool waitUntilReached = false, int waitTimeout = 0)`**11.10.3.38** `void CKatana::moveMotorBy (short number, double radianAngle, bool waitUntilReached = false, int waitTimeout = 0)`**11.10.3.39** `void CKatana::moveMotorToEnc (short number, int encoders, bool waitUntilReached = false, int encTolerance = 100, int waitTimeout = 0)`**11.10.3.40** `void CKatana::moveMotorTo (short number, double radianAngle, bool waitUntilReached = false, int waitTimeout = 0)`**11.10.3.41** `void CKatana::waitForMotor (short number, int encoders, int encTolerance = 100, short mode = 0, int waitTimeout = 5000)`**11.10.3.42** `void CKatana::moveRobotToEnc (std::vector< int >::const_iterator start, std::vector< int >::const_iterator end, bool waitUntilReached = false, int encTolerance = 100, int waitTimeout = 0)`

Move to robot to given encoders.

You can provide less values than the number of motors. In that case only the given ones will be moved. This can be usefull in cases where you want to move the robot but you don't want to move the gripper.

11.10.3.43 `void CKatana::moveRobotToEnc (std::vector< int > encoders, bool waitUntilReached = false, int encTolerance = 100, int waitTimeout = 0)`

Move to robot to given encoders in the vector-container.

This method is mainly provided for convenience. Catch by value (and not by reference) is intended to avoid nasty wrapping code.

11.10.3.44 void CKatana::moveRobotToEnc4D (std::vector< int > *target*, int *velocity* = 180, int *acceleration* = 1, int *encTolerance* = 100)

Move to robot to given target in the vector-container with the given velocity, acceleration and tolerance.

11.10.3.45 void CKatana::openGripper (bool *waitUntilReached* = false, int *waitTimeout* = 100)

11.10.3.46 void CKatana::closeGripper (bool *waitUntilReached* = false, int *waitTimeout* = 100)

11.10.3.47 void CKatana::freezeRobot ()

11.10.3.48 void CKatana::freezeMotor (short *number*)

11.10.3.49 void CKatana::switchRobotOn ()

11.10.3.50 void CKatana::switchRobotOff ()

11.10.3.51 void CKatana::switchMotorOn (short *number*)

11.10.3.52 void CKatana::switchMotorOff (short *number*)

11.10.3.53 void CKatana::startSplineMovement (bool *exactflag*, int *moreflag* = 1)

Start a spline movement.

Parameters:

exactflag Set it to true if you want the position controller activated after the movement

moreflag 0 = start moving more following, 1 = last or a single polynomial movement, 2 = do not start moving yet more following

11.10.3.54 void CKatana::startFourSplinesMovement (bool *exactflag*)

Start a fourSplines movement.

Parameters:

exactflag Set it to true if you want the position controller activated after the movement

11.10.3.55 void CKatana::sendSplineToMotor (unsigned short *number*, short *targetPosition*, short *duration*, short *p1*, short *p2*, short *p3*, short *p4*)

Send one spline to the motor.

Parameters:

duration Duration has to be given in 10ms units

11.10.3.56 void CKatana::sendFourSplinesToMotor (unsigned short *number*, short *targetPosition*, short *duration*, std::vector< short > & *coefficients*)

Send four splines to the motor.

Parameters:

duration Duration has to be given in 10ms units

coefficients 4x4 coefficients have to be passed or the function will cause an assertion.

11.10.3.57 void CKatana::sendFourSplinesToMotor (unsigned short *number*, short *targetPosition*, short *duration*, short *p01*, short *p11*, short *p21*, short *p31*, short *p02*, short *p12*, short *p22*, short *p32*, short *p03*, short *p13*, short *p23*, short *p33*, short *p04*, short *p14*, short *p24*, short *p34*)

11.10.4 Member Data Documentation

11.10.4.1 CKatBase* CKatana::base [protected]

base katana

Definition at line 67 of file kmlExt.h.

11.10.4.2 bool CKatana::_gripperIsPresent [protected]

Definition at line 69 of file kmlExt.h.

11.10.4.3 int CKatana::_gripperOpenEncoders [protected]

Definition at line 70 of file kmlExt.h.

11.10.4.4 int CKatana::_gripperCloseEncoders [protected]

Definition at line 71 of file kmlExt.h.

11.10.4.5 int CKatana::mKatanaType [protected]

The type of KatanaXXX (300 or 400).

Definition at line 73 of file kmlExt.h.

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

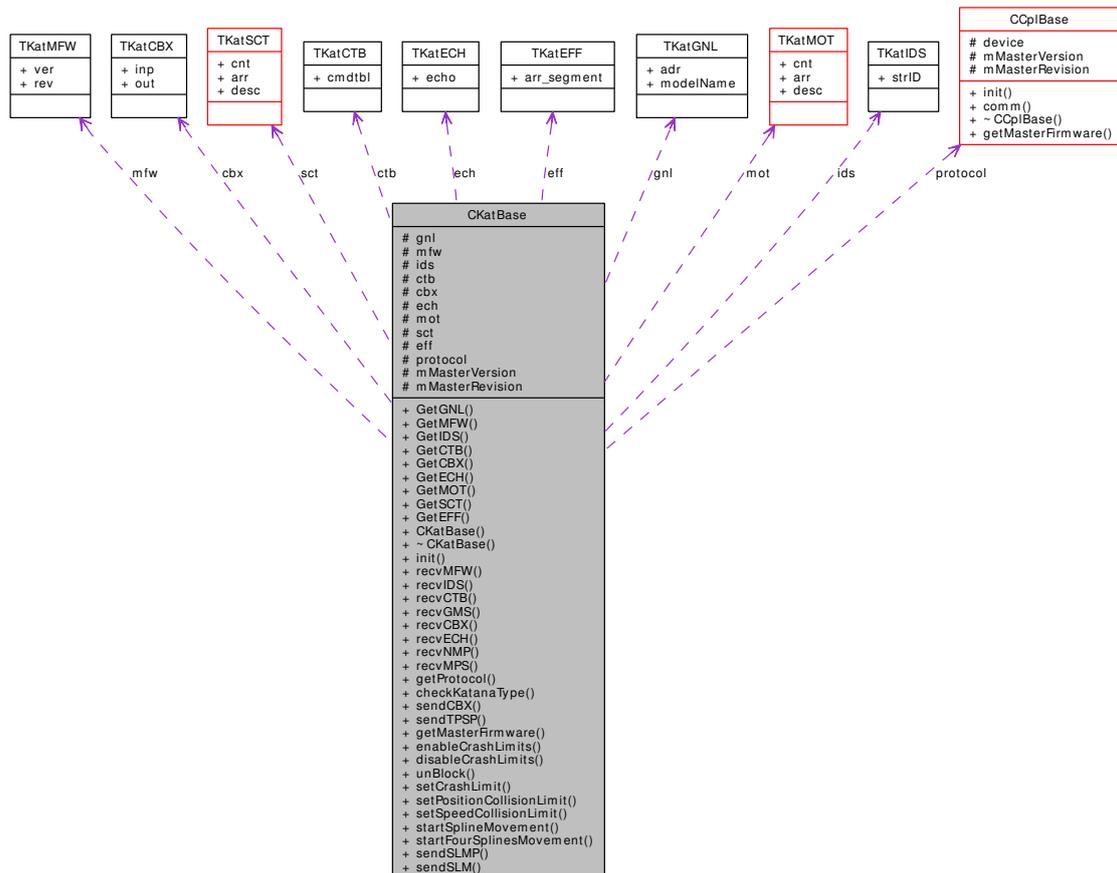
- include/KNI/kmlExt.h

11.11 CKatBase Class Reference

Base Katana class.

```
#include <kmlBase.h>
```

Collaboration diagram for CKatBase:



Public Member Functions

- const [TKatGNL](#) * [GetGNL](#) ()
Get a pointer to the desired structure.
- const [TKatMFW](#) * [GetMFW](#) ()
Get a pointer to the desired structure.
- const [TKatIDS](#) * [GetIDS](#) ()
Get a pointer to the desired structure.
- const [TKatCTB](#) * [GetCTB](#) ()
Get a pointer to the desired structure.
- const [TKatCBX](#) * [GetCBX](#) ()

Get a pointer to the desired structure.

- const [TKatECH](#) * [GetECH](#) ()

Get a pointer to the desired structure.

- const [TKatMOT](#) * [GetMOT](#) ()

Get a pointer to the desired structure.

- const [TKatSCT](#) * [GetSCT](#) ()

Get a pointer to the desired structure.

- [TKatEFF](#) * [GetEFF](#) ()

Get a pointer to the desired structure.

- [CKatBase](#) ()

- virtual [~CKatBase](#) ()

destructor

- virtual bool [init](#) (const [TKatGNL](#) _gnl, const [TKatMOT](#) _mot, const [TKatSCT](#) _sct, const [TKatEFF](#) _eff, [CCplBase](#) *_protocol)

- void [recvMFW](#) ()

receive data

- void [recvIDS](#) ()

receive data

- void [recvCTB](#) ()

receive data

- void [recvGMS](#) ()

receive data

- void [recvCBX](#) ()

receive data

- void [recvECH](#) ()

receive data

- void [recvNMP](#) ()

receive data

- void [recvMPS](#) ()

read all motor positions simultaneously

- [CCplBase](#) * [getProtocol](#) ()

get a handle of the protocol, used in [CKatana](#)

- int [checkKatanaType](#) (int type)

checks for a K300 or K400

- void [sendCBX](#) (const [TKatCBX](#) *_cbx)
send data
- void [sendTPSP](#) ()
parallel movements
- void [getMasterFirmware](#) (short *fw, short *rev)
Get the master firmware of the robot we are communicating with.
- void [enableCrashLimits](#) ()
crash limits enable
- void [disableCrashLimits](#) ()
crash limits disable
- void [unBlock](#) ()
unblock robot after a crash
- void [setCrashLimit](#) (long idx, int limit)
set collision limits
- void [setPositionCollisionLimit](#) (long idx, int limit)
set collision position limits
- void [setSpeedCollisionLimit](#) (long idx, int limit)
set collision speed limits
- void [startSplineMovement](#) (bool exactflag, int moreflag=1)
Start a spline movement.
- void [startFourSplinesMovement](#) (bool exactflag)
Start a fourSplines movement.
- void [sendSLMP](#) (byte *p)
linear movements
- void [sendSLM](#) (bool exactflag)
linear movements

Protected Attributes

- [TKatGNL](#) gnl
katana general
- [TKatMFW](#) mfw
master's firmware version/revision
- [TKatIDS](#) ids

ID string.

- [TKatCTB ctb](#)

cmd table

- [TKatCBX cbx](#)

connector box

- [TKatECH ech](#)

echo

- [TKatMOT mot](#)

motors

- [TKatSCT sct](#)

sensor controllers

- [TKatEFF eff](#)

end effector

- [CCplBase * protocol](#)

protocol interface

- short [mMasterVersion](#)

master version of robot we are communicating with

- short [mMasterRevision](#)

master firmware revision

11.11.1 Detailed Description

Base Katana class.

This class is the main object controlling the whole katana; to use it, it has to be initialized by using its `init` function; those function expects a initialized protocol class, which in turn expects an initialized device! after the initialization, it does not mean that the coordinates (encoder values) of the motors have been set correctly; for that a calibration is needed; that calibration can be executed either by using the [CKatana](#) class in the 'kmlExt' module (which encapsulates this class) or by writing your own calibrations function..

Definition at line 132 of file `kmlBase.h`.

11.11.2 Constructor & Destructor Documentation

11.11.2.1 `CKatBase::CKatBase()` [inline]

Definition at line 172 of file `kmlBase.h`.

11.11.2.2 virtual CKatBase::~~CKatBase () [inline, virtual]

destructor

Definition at line 175 of file kmlBase.h.

11.11.3 Member Function Documentation**11.11.3.1 const TKatGNL* CKatBase::GetGNL ()** [inline]

Get a pointer to the desired structure.

Definition at line 152 of file kmlBase.h.

11.11.3.2 const TKatMFW* CKatBase::GetMFW () [inline]

Get a pointer to the desired structure.

Definition at line 154 of file kmlBase.h.

11.11.3.3 const TKatIDS* CKatBase::GetIDS () [inline]

Get a pointer to the desired structure.

Definition at line 156 of file kmlBase.h.

11.11.3.4 const TKatCTB* CKatBase::GetCTB () [inline]

Get a pointer to the desired structure.

Definition at line 158 of file kmlBase.h.

11.11.3.5 const TKatCBX* CKatBase::GetCBX () [inline]

Get a pointer to the desired structure.

Definition at line 160 of file kmlBase.h.

11.11.3.6 const TKatECH* CKatBase::GetECH () [inline]

Get a pointer to the desired structure.

Definition at line 162 of file kmlBase.h.

11.11.3.7 const TKatMOT* CKatBase::GetMOT () [inline]

Get a pointer to the desired structure.

Definition at line 165 of file kmlBase.h.

11.11.3.8 `const TKatSCT* CKatBase::GetSCT () [inline]`

Get a pointer to the desired structure.

Definition at line 167 of file kmlBase.h.

11.11.3.9 `TKatEFF* CKatBase::GetEFF () [inline]`

Get a pointer to the desired structure.

Definition at line 169 of file kmlBase.h.

11.11.3.10 `virtual bool CKatBase::init (const TKatGNL _gnl, const TKatMOT _mot, const TKatSCT _sct, const TKatEFF _eff, CCplBase * _protocol) [virtual]`**Parameters:**

_gnl general attributes

_mot motor attributes

_sct sensor controller attributes

_eff end effector attributes

_protocol desired protocol

11.11.3.11 `void CKatBase::recvMFW ()`

receive data

11.11.3.12 `void CKatBase::recvIDS ()`

receive data

11.11.3.13 `void CKatBase::recvCTB ()`

receive data

11.11.3.14 `void CKatBase::recvGMS ()`

receive data

11.11.3.15 `void CKatBase::recvCBX ()`

receive data

11.11.3.16 `void CKatBase::recvECH ()`

receive data

11.11.3.17 void CKatBase::recvNMP ()

receive data

11.11.3.18 void CKatBase::recvMPS ()

read all motor positions simultaneously

11.11.3.19 CCplBase* CKatBase::getProtocol () [inline]

get a handle of the protocol, used in [CKatana](#)

Definition at line 202 of file kmlBase.h.

11.11.3.20 int CKatBase::checkKatanaType (int type)

checks for a K300 or K400

11.11.3.21 void CKatBase::sendCBX (const TKatCBX * _cbx)

send data

11.11.3.22 void CKatBase::sendTPSP ()

parallel movements

deprecated: for use with old Katana5M only

11.11.3.23 void CKatBase::getMasterFirmware (short * fw, short * rev)

Get the master firmware of the robot we are communicating with.

Get master firmware read at initialization time.

11.11.3.24 void CKatBase::enableCrashLimits ()

crash limits enable

11.11.3.25 void CKatBase::disableCrashLimits ()

crash limits disable

11.11.3.26 void CKatBase::unBlock ()

unblock robot after a crash

11.11.3.27 void CKatBase::setCrashLimit (long *idx*, int *limit*)

set collision limits

//deprecated, use speed & position

11.11.3.28 void CKatBase::setPositionCollisionLimit (long *idx*, int *limit*)

set collision position limits

11.11.3.29 void CKatBase::setSpeedCollisionLimit (long *idx*, int *limit*)

set collision speed limits

11.11.3.30 void CKatBase::startSplineMovement (bool *exactflag*, int *moreflag* = 1)

Start a spline movement.

Parameters:

exactflag Set it to true if you want the position controller activated after the movement

moreflag 0 = start moving more following, 1 = last or a single polynomial movement, 2 = do not start moving yet more following

11.11.3.31 void CKatBase::startFourSplinesMovement (bool *exactflag*)

Start a fourSplines movement.

Parameters:

exactflag Set it to true if you want the position controller activated after the movement

11.11.3.32 void CKatBase::sendSLMP (byte * *p*)

linear movements

11.11.3.33 void CKatBase::sendSLM (bool *exactflag*)

linear movements

11.11.4 Member Data Documentation**11.11.4.1 [TKatGNL CKatBase::gnl](#) [protected]**

katana general

Definition at line 135 of file kmlBase.h.

Referenced by CScBase::GetGNL(), and CMotBase::GetGNL().

11.11.4.2 [TKatMFW CKatBase::mfw](#) [protected]

master's firmware version/revision

Definition at line 136 of file kmlBase.h.

11.11.4.3 [TKatIDS CKatBase::ids](#) [protected]

ID string.

Definition at line 137 of file kmlBase.h.

11.11.4.4 [TKatCTB CKatBase::ctb](#) [protected]

cmd table

Definition at line 138 of file kmlBase.h.

11.11.4.5 [TKatCBX CKatBase::cbx](#) [protected]

connector box

Definition at line 139 of file kmlBase.h.

11.11.4.6 [TKatECH CKatBase::ech](#) [protected]

echo

Definition at line 140 of file kmlBase.h.

11.11.4.7 [TKatMOT CKatBase::mot](#) [protected]

motors

Definition at line 142 of file kmlBase.h.

11.11.4.8 [TKatSCT CKatBase::sct](#) [protected]

sensor controllers

Definition at line 143 of file kmlBase.h.

11.11.4.9 [TKatEFF CKatBase::eff](#) [protected]

end effector

Definition at line 144 of file kmlBase.h.

11.11.4.10 [CCplBase* CKatBase::protocol](#) [protected]

protocol interface

Definition at line 146 of file kmlBase.h.

11.11.4.11 short **CKatBase::mMasterVersion** [protected]

master version of robot we are communicating with

Definition at line 147 of file kmlBase.h.

11.11.4.12 short **CKatBase::mMasterRevision** [protected]

master firmware revision

Definition at line 148 of file kmlBase.h.

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

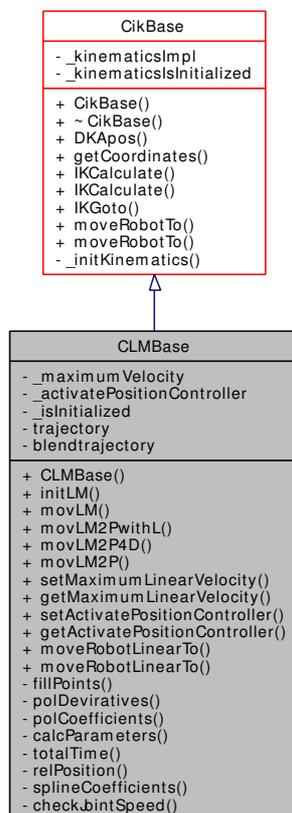
- [include/KNI/kmlBase.h](#)

11.12 CLMBase Class Reference

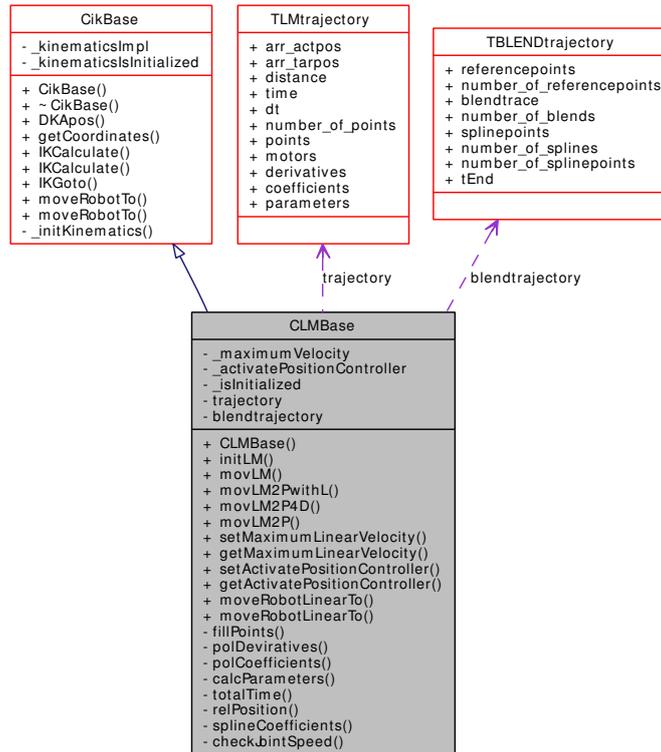
Linear movement Class.

```
#include <lmBase.h>
```

Inheritance diagram for CLMBase:



Collaboration diagram for CLMBase:



Public Member Functions

- [CLMBase \(\)](#)
- void [initLM \(\)](#)

Initialize the parameters for the linear movements.

- void [movLM](#) (double X, double Y, double Z, double A1, double Be, double Ga, bool exactflag, double vmax, bool wait=true, int tolerance=100, long timeout=TM_ENDLESS)

Parameters:

wait has to be true with new implementation of movLM2P

- void [movLM2PwithL](#) (double X1, double Y1, double Z1, double A11, double Be1, double Ga1, double X2, double Y2, double Z2, double A12, double Be2, double Ga2, bool exactflag, double vmax, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)

Old version of movLM2P which uses L-Command (only 4 splines).

- void [movLM2P4D](#) (double X1, double Y1, double Z1, double A11, double Be1, double Ga1, double X2, double Y2, double Z2, double A12, double Be2, double Ga2, bool exactflag, double vmax, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
- void [movLM2P](#) (double X1, double Y1, double Z1, double A11, double Be1, double Ga1, double X2, double Y2, double Z2, double A12, double Be2, double Ga2, bool exactflag, double vmax, bool wait=true, int tolerance=100, long timeout=TM_ENDLESS)

New version of movLM2P with multiple splines.

- void [setMaximumLinearVelocity](#) (double maximumVelocity)
- double [getMaximumLinearVelocity \(\)](#) const

- void [setActivatePositionController](#) (bool activate)
Re-Activate the position controller after the linear movement.
- bool [getActivatePositionController](#) ()
Check if the position controller will be activated after the linear movement.
- void [moveRobotLinearTo](#) (double x, double y, double z, double phi, double theta, double psi, bool waitUntilReached=true, int waitTimeout=TM_ENDLESS)
Parameters:
waitUntilReached has to be true with new implementation of movLM2P
- void [moveRobotLinearTo](#) (std::vector< double > coordinates, bool waitUntilReached=true, int waitTimeout=TM_ENDLESS)
This method does the same as the one above and is mainly provided for convenience.

Private Member Functions

- void [fillPoints](#) (double vmax)
- void [polDeviratives](#) ()
- void [polCoefficients](#) ()
- void [calcParameters](#) (double *arr_actpos, double *arr_tarpos, double vmax)
- double [totalTime](#) (double distance, double acc, double dec, double vmax)
Calculates time needed for movement over a distance.
- double [relPosition](#) (double reltime, double distance, double acc, double dec, double vmax)
Calculates the relative position reached after the relative time given.
- void [splineCoefficients](#) (int steps, double *timearray, double *encoderarray, double *arr_p1, double *arr_p2, double *arr_p3, double *arr_p4)
Calculates the spline coefficient and stores them in arr_p1 - arr_p4.
- bool [checkJointSpeed](#) (std::vector< int > lastsolution, std::vector< int > solution, double time)
Checks if the joint speeds are below speed limit.

Private Attributes

- double [_maximumVelocity](#)
- bool [_activatePositionController](#)
- bool [_isInitialized](#)
- [TLMtrajectory](#) trajectory
- [TBLENDtrajectory](#) blendtrajectory

11.12.1 Detailed Description

Linear movement Class.

This class allows to do linear movements with the Katana robot.

Definition at line 153 of file lmBase.h.

11.12.2 Constructor & Destructor Documentation

11.12.2.1 CLMBase::CLMBase () [inline]

Definition at line 239 of file lmBase.h.

11.12.3 Member Function Documentation

11.12.3.1 void CLMBase::fillPoints (double *vmax*) [private]

11.12.3.2 void CLMBase::polDeviratives () [private]

11.12.3.3 void CLMBase::polCoefficients () [private]

11.12.3.4 void CLMBase::calcParameters (double * *arr_actpos*, double * *arr_tarpos*, double *vmax*) [private]

11.12.3.5 double CLMBase::totalTime (double *distance*, double *acc*, double *dec*, double *vmax*) [private]

Calculates time needed for movement over a distance.

Author:

Jonas Haller

Parameters:

distance distance of the movement in mm
acc acceleration at the beginning in mm/s²
dec deceleration at the end in mm/s²
vmax maximum velocity of the movement in mm/s

Returns:

time needed for the movement in s

11.12.3.6 double CLMBase::relPosition (double *reltime*, double *distance*, double *acc*, double *dec*, double *vmax*) [private]

Calculates the relative position reached after the relative time given.

Author:

Jonas Haller

Parameters:

reltime relative time (fraction of totaltime)
distance distance of the movement in mm
acc acceleration at the beginning in mm/s²
dec deceleration at the end in mm/s²

vmax maximum velocity of the movement in mm/s

Returns:

relative distance (fraction of distance)

11.12.3.7 void CLMBase::splineCoefficients (int steps, double * timearray, double * encoderarray, double * arr_p1, double * arr_p2, double * arr_p3, double * arr_p4) [private]

Calculates the spline coefficient and stores them in arr_p1 - arr_p4.

Boundary conditions are that $f_1' = 0$ and $f_n' = 0$ (zero velocity at beginning and end of the movement) and $f_i'' = P_{(i+1)}$.

Author:

Jonas Haller

Parameters:

steps number of splines to calculate

timearray times of the points (length = steps + 1)

encoderarray encoder values of the points (length = steps + 1)

arr_p1 to return parameters 1 (length = steps)

arr_p2 to return parameters 2 (length = steps)

arr_p3 to return parameters 3 (length = steps)

arr_p4 to return parameters 4 (length = steps)

Returns:

void

11.12.3.8 bool CLMBase::checkJointSpeed (std::vector< int > lastsolution, std::vector< int > solution, double time) [private]

Checks if the joint speeds are below speed limit.

Maximum joint speed is 180enc / 10ms.

Author:

Jonas Haller

Parameters:

lastsolution encoder values of last point

solution encoder values of current point

time time difference between the points in s

Returns:

true if joint speeds ok, false if joint speed too high

11.12.3.9 void CLMBase::initLM ()

Initialize the parameters for the linear movements.

This is in the case you want to initialize it manually

Note:

If you do not call it, [moveRobotLinearTo\(\)](#) will do it for you automatically

11.12.3.10 void CLMBase::movLM (double X, double Y, double Z, double A1, double Be, double Ga, bool exactflag, double vmax, bool wait = true, int tolerance = 100, long timeout = TM_ENDLESS)**Parameters:**

wait has to be true with new implementation of movLM2P

11.12.3.11 void CLMBase::movLM2PwithL (double X1, double Y1, double Z1, double A11, double Be1, double Ga1, double X2, double Y2, double Z2, double A12, double Be2, double Ga2, bool exactflag, double vmax, bool wait = false, int tolerance = 100, long timeout = TM_ENDLESS)

Old version of movLM2P which uses L-Command (only 4 splines).

11.12.3.12 void CLMBase::movLM2P4D (double X1, double Y1, double Z1, double A11, double Be1, double Ga1, double X2, double Y2, double Z2, double A12, double Be2, double Ga2, bool exactflag, double vmax, bool wait = false, int tolerance = 100, long timeout = TM_ENDLESS)**11.12.3.13 void CLMBase::movLM2P (double X1, double Y1, double Z1, double A11, double Be1, double Ga1, double X2, double Y2, double Z2, double A12, double Be2, double Ga2, bool exactflag, double vmax, bool wait = true, int tolerance = 100, long timeout = TM_ENDLESS)**

New version of movLM2P with multiple splines.

Author:

Jonas Haller

Parameters:

X1 X coordinate of actual position

Y1 Y coordinate of actual position

Z1 Z coordinate of actual position

Ph1 Phi angle of actual position

Th1 Theta angle of actual position

Ps1 Psi angle of actual position

X2 X coordinate of target position

Y2 Y coordinate of target position

Z2 Z coordinate of target position
Ph2 Phi angle of target position
Th2 Theta angle of target position
Ps2 Psi angle of target position
exactflag activate the position controller after the movement
vmax maximum velocity of the movement in mm/s
wait param for legacy reasons only, has to be true
tolerance tolerance for all motor encoders
timeout timeout for linear movement in ms

Exceptions:

NoSolutionException if no solution found for IK
JointSpeedException if joint speed too high
WaitParameterException if wait set to false

Returns:

void

11.12.3.14 void CLMBase::setMaximumLinearVelocity (double *maximumVelocity*)

11.12.3.15 double CLMBase::getMaximumLinearVelocity () const

11.12.3.16 void CLMBase::setActivatePositionController (bool *activate*)

Re-Activate the position controller after the linear movement.

Note:

This can result in a small movement after the movement

11.12.3.17 bool CLMBase::getActivatePositionController ()

Check if the position controller will be activated after the linear movement.

11.12.3.18 void CLMBase::moveRobotLinearTo (double *x*, double *y*, double *z*, double *phi*, double *theta*, double *psi*, bool *waitUntilReached* = true, int *waitTimeout* = TM_ENDLESS)

Parameters:

waitUntilReached has to be true with new implementation of movLM2P

11.12.3.19 `void CLMBase::moveRobotLinearTo (std::vector< double > coordinates, bool waitUntilReached = true, int waitTimeout = TM_ENDLESS)`

This method does the same as the one above and is mainly provided for convenience.

Note:

You can call this function in python using tuples: Example: `katana.moveRobotLinearTo(x,y,z,phi,theta,psi)`

If the size of the container is smaller than 6, it will throw an exception!

11.12.4 Member Data Documentation

11.12.4.1 `double CLMBase::_maximumVelocity` [private]

Definition at line 156 of file `lmBase.h`.

11.12.4.2 `bool CLMBase::_activatePositionController` [private]

Definition at line 157 of file `lmBase.h`.

11.12.4.3 `bool CLMBase::_isInitialized` [private]

Definition at line 158 of file `lmBase.h`.

11.12.4.4 `TLMtrajectory CLMBase::trajectory` [private]

Definition at line 161 of file `lmBase.h`.

11.12.4.5 `TBLENDtrajectory CLMBase::blendtrajectory` [private]

Definition at line 162 of file `lmBase.h`.

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

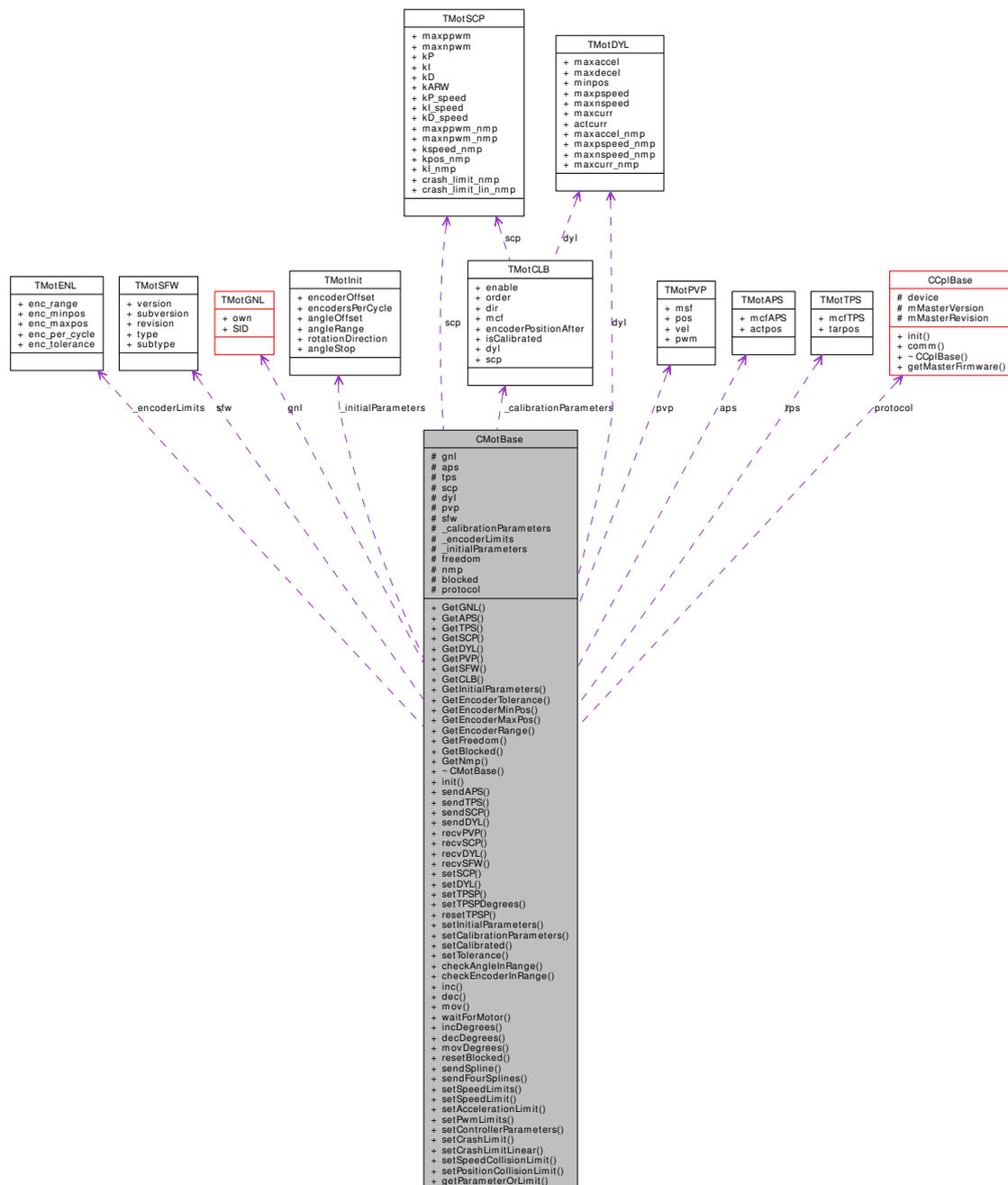
- `include/KNI_LM/lmBase.h`

11.13 CMotBase Class Reference

Motor class.

```
#include <kmlMotBase.h>
```

Collaboration diagram for CMotBase:



Public Member Functions

- const [TMotGNL](#) * [GetGNL](#) ()
- const [TMotAPS](#) * [GetAPS](#) ()
- const [TMotTPS](#) * [GetTPS](#) ()
- const [TMotSCP](#) * [GetSCP](#) ()
- const [TMotDYL](#) * [GetDYL](#) ()
- const [TMotPVP](#) * [GetPVP](#) ()
- const [TMotSFW](#) * [GetSFW](#) ()
- const [TMotCLB](#) * [GetCLB](#) ()
- const [TMotInit](#) * [GetInitialParameters](#) ()
- const int [GetEncoderTolerance](#) ()
- const int [GetEncoderMinPos](#) ()
Returns the min Position of the Encoder.

- const int [GetEncoderMaxPos](#) ()
Returns the max Position of the Encoder.

- const int [GetEncoderRange](#) ()
Returns Encoder Range of the Encoder.

- const bool [GetFreedom](#) ()
Get the value of the freedom property.

- const bool [GetBlocked](#) ()
Get the value of the blocked property.

- const bool [GetNmp](#) ()
Get the value of the nmp property.

- virtual [~CMotBase](#) ()
- bool [init](#) ([CKatBase](#) *_own, const [TMotDesc](#) _motDesc, [CCplBase](#) *protocol)
- void [sendAPS](#) (const [TMotAPS](#) *_aps)
send data

- void [sendTPS](#) (const [TMotTPS](#) *_tps)
send data

- void [sendSCP](#) (const [TMotSCP](#) *_scp)
send data

- void [sendDYL](#) (const [TMotDYL](#) *_dyl)
send data

- void [recvPVP](#) ()
receive data

- void [recvSCP](#) ()
receive data

- void [recvDYL](#) ()
receive data
- void [recvSFW](#) ()
receive data
- void [setSCP](#) (TMotSCP _scp)
- void [setDYL](#) (TMotDYL _dyl)
- void [setTPSP](#) (int tar)
parallel movement
- void [setTPSPDegrees](#) (double tar)
- void [resetTPSP](#) ()
parallel movement
- void [setInitialParameters](#) (double angleOffset, double angleRange, int encodersPerCycle, int encoderOffset, int rotationDirection)
- void [setCalibrationParameters](#) (bool doCalibration, short order, TSearchDir direction, TMotCmdFlg motorFlagAfter, int encoderPositionAfter)
- void [setCalibrated](#) (bool calibrated)
- void [setTolerance](#) (int tolerance)
- bool [checkAngleInRange](#) (double angle)
check limits in encoder values
- bool [checkEncoderInRange](#) (int encoder)
- void [inc](#) (int dif, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Increments the motor specified by an index postion in encoder units.
- void [dec](#) (int dif, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Decrements the motor specified by an index postion in encoder units.
- void [mov](#) (int tar, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Moves the motor specified by an index to a given target position in encoder units.
- void [waitForMotor](#) (int tar, int encTolerance=100, short mode=0, int waitTimeout=TM_ENDLESS)
Waits until the Motor has reached the given targen position.
- void [incDegrees](#) (double dif, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Increments the motor specified by an index postion in degrees.
- void [decDegrees](#) (double dif, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Decrements the motor specified by an index postion in degrees.
- void [movDegrees](#) (double tar, bool wait=false, int tolerance=100, long timeout=TM_ENDLESS)
Moves the motor specified by an index to a given target position in degrees.
- void [resetBlocked](#) ()
unblock the motor.

- void [sendSpline](#) (short targetPosition, short duration, short p1, short p2, short p3, short p4)
Send one spline to the motor.
- void [sendFourSplines](#) (short targetPosition, short duration, std::vector< short > &coefficients)
Send four splines to the motor.
- void [setSpeedLimits](#) (short positiveVelocity, short negativeVelocity)
Set speed limits.
- void [setSpeedLimit](#) (short velocity)
- void [setAccelerationLimit](#) (short acceleration)
Set the acceleration limits.
- void [setPwmLimits](#) (byte maxppwm, byte maxnpwm)
Set the PWM limits.
- void [setControllerParameters](#) (byte kSpeed, byte kPos, byte kI)
Set the controller parameters.
- void [setCrashLimit](#) (int limit)
Set the crash limit.
- void [setCrashLimitLinear](#) (int limit_lin)
Set the crash limit linear.
- void [setSpeedCollisionLimit](#) (int limit)
Set the collision limit.
- void [setPositionCollisionLimit](#) (int limit)
Set the collision limit.
- void [getParameterOrLimit](#) (int subcommand, byte *R1, byte *R2, byte *R3)
Get parameters or limits.

Protected Attributes

- [TMotGNL gnl](#)
motor generals
- [TMotAPS aps](#)
actual position
- [TMotTPS tps](#)
target position
- [TMotSCP scp](#)
static controller parameters

- [TMotDYL dyl](#)
dynamic limits
- [TMotPVP pvp](#)
reading motor parameters
- [TMotSFW sfw](#)
slave firmware
- [TMotCLB _calibrationParameters](#)
calibration structure
- [TMotENL _encoderLimits](#)
motor limits in encoder values
- [TMotInit _initialParameters](#)
- bool [freedom](#)
if it is set, it will move on a parallel movement
- bool [nmp](#)
true if new motor parameters are implemented on the firmware
- bool [blocked](#)
true if the motor was blocked due to a crash of the robot
- [CCplBase](#) * [protocol](#)
protocol interface

Friends

- class [CKatBase](#)

11.13.1 Detailed Description

Motor class.

This class allows to control one motor; to control a motor it has to be initialized by using the init function. And the usage the internal allocated resources should be deallocated by using the 'free' method.

Definition at line 219 of file kmlMotBase.h.

11.13.2 Constructor & Destructor Documentation

11.13.2.1 virtual [CMotBase::~~CMotBase](#) () [inline, virtual]

Definition at line 267 of file kmlMotBase.h.

11.13.3 Member Function Documentation

11.13.3.1 `const TMotGNL* CMotBase::GetGNL ()` [inline]

Definition at line 241 of file kmlMotBase.h.

References CKatBase::gnl.

11.13.3.2 `const TMotAPS* CMotBase::GetAPS ()` [inline]

Definition at line 242 of file kmlMotBase.h.

11.13.3.3 `const TMotTPS* CMotBase::GetTPS ()` [inline]

Definition at line 243 of file kmlMotBase.h.

11.13.3.4 `const TMotSCP* CMotBase::GetSCP ()` [inline]

Definition at line 244 of file kmlMotBase.h.

11.13.3.5 `const TMotDYL* CMotBase::GetDYL ()` [inline]

Definition at line 245 of file kmlMotBase.h.

11.13.3.6 `const TMotPVP* CMotBase::GetPVP ()` [inline]

Definition at line 246 of file kmlMotBase.h.

11.13.3.7 `const TMotSFW* CMotBase::GetSFW ()` [inline]

Definition at line 247 of file kmlMotBase.h.

11.13.3.8 `const TMotCLB* CMotBase::GetCLB ()` [inline]

Definition at line 248 of file kmlMotBase.h.

11.13.3.9 `const TMotInit* CMotBase::GetInitialParameters ()` [inline]

Definition at line 250 of file kmlMotBase.h.

11.13.3.10 `const int CMotBase::GetEncoderTolerance ()` [inline]

Definition at line 251 of file kmlMotBase.h.

11.13.3.11 `const int CMotBase::GetEncoderMinPos ()` [inline]

Returns the min Position of the Encoder.

Definition at line 252 of file kmlMotBase.h.

11.13.3.12 `const int CMotBase::GetEncoderMaxPos ()` [inline]

Returns the max Position of the Encoder.

Definition at line 253 of file kmlMotBase.h.

11.13.3.13 `const int CMotBase::GetEncoderRange ()` [inline]

Returns Encoder Range of the Encoder.

Definition at line 254 of file kmlMotBase.h.

11.13.3.14 `const bool CMotBase::GetFreedom ()` [inline]

Get the value of the freedom property.

Definition at line 257 of file kmlMotBase.h.

11.13.3.15 `const bool CMotBase::GetBlocked ()` [inline]

Get the value of the blocked property.

Definition at line 259 of file kmlMotBase.h.

11.13.3.16 `const bool CMotBase::GetNmp ()` [inline]

Get the value of the nmp property.

Definition at line 261 of file kmlMotBase.h.

11.13.3.17 `bool CMotBase::init (CKatBase * _own, const TMotDesc _motDesc, CCplBase * protocol)`**11.13.3.18** `void CMotBase::sendAPS (const TMotAPS * _aps)`

send data

11.13.3.19 `void CMotBase::sendTPS (const TMotTPS * _tps)`

send data

11.13.3.20 `void CMotBase::sendSCP (const TMotSCP * _scp)`

send data

11.13.3.21 void CMotBase::sendDYL (const [TMotDYL](#) * *_dyl*)

send data

11.13.3.22 void CMotBase::recvPVP ()

receive data

11.13.3.23 void CMotBase::recvSCP ()

receive data

11.13.3.24 void CMotBase::recvDYL ()

receive data

11.13.3.25 void CMotBase::recvSFW ()

receive data

11.13.3.26 void CMotBase::setSCP ([TMotSCP](#) *_scp*) [inline]

Definition at line 289 of file kmlMotBase.h.

11.13.3.27 void CMotBase::setDYL ([TMotDYL](#) *_dyl*) [inline]

Definition at line 290 of file kmlMotBase.h.

11.13.3.28 void CMotBase::setTPSP (int *tar*)

parallel movement

deprecated: for use with old Katana5M only

11.13.3.29 void CMotBase::setTPSPDegrees (double *tar*)**11.13.3.30 void CMotBase::resetTPSP ()**

parallel movement

deprecated: for use with old Katana5M only

11.13.3.31 void CMotBase::setInitialParameters (double *angleOffset*, double *angleRange*, int *encodersPerCycle*, int *encoderOffset*, int *rotationDirection*)

11.13.3.32 void CMotBase::setCalibrationParameters (bool *doCalibration*, short *order*, TSearchDir *direction*, TMotCmdFlg *motorFlagAfter*, int *encoderPositionAfter*)

11.13.3.33 void CMotBase::setCalibrated (bool *calibrated*)

11.13.3.34 void CMotBase::setTolerance (int *tolerance*)

11.13.3.35 bool CMotBase::checkAngleInRange (double *angle*)

check limits in encoder values

11.13.3.36 bool CMotBase::checkEncoderInRange (int *encoder*)

11.13.3.37 void CMotBase::inc (int *dif*, bool *wait* = false, int *tolerance* = 100, long *timeout* = TM_ENDLESS)

Increments the motor specified by an index postion in encoder units.

11.13.3.38 void CMotBase::dec (int *dif*, bool *wait* = false, int *tolerance* = 100, long *timeout* = TM_ENDLESS)

Decrements the motor specified by an index postion in encoder units.

11.13.3.39 void CMotBase::mov (int *tar*, bool *wait* = false, int *tolerance* = 100, long *timeout* = TM_ENDLESS)

Moves the motor specified by an index to a given target position in encoder units.

11.13.3.40 void CMotBase::waitForMotor (int *tar*, int *encTolerance* = 100, short *mode* = 0, int *waitTimeout* = TM_ENDLESS)

Waits until the Motor has reached the given targen position.

11.13.3.41 void CMotBase::incDegrees (double *dif*, bool *wait* = false, int *tolerance* = 100, long *timeout* = TM_ENDLESS)

Increments the motor specified by an index postion in degrees.

11.13.3.42 void CMotBase::decDegrees (double *dif*, bool *wait* = false, int *tolerance* = 100, long *timeout* = TM_ENDLESS)

Decrements the motor specified by an index postion in degrees.

11.13.3.43 void CMotBase::movDegrees (double *tar*, bool *wait* = false, int *tolerance* = 100, long *timeout* = TM_ENDLESS)

Moves the motor specified by an index to a given target position in degrees.

11.13.3.44 void CMotBase::resetBlocked ()

unlock the motor.

11.13.3.45 void CMotBase::sendSpline (short *targetPosition*, short *duration*, short *p1*, short *p2*, short *p3*, short *p4*)

Send one spline to the motor.

Parameters:

duration Duration has to be given in 10ms units

11.13.3.46 void CMotBase::sendFourSplines (short *targetPosition*, short *duration*, std::vector<short > & *coefficients*)

Send four splines to the motor.

Parameters:

duration Duration has to be given in 10ms units

coefficients 4x4 coefficients have to be passed or the function will cause an assertion.

11.13.3.47 void CMotBase::setSpeedLimits (short *positiveVelocity*, short *negativeVelocity*)

Set speed limits.

11.13.3.48 void CMotBase::setSpeedLimit (short *velocity*) [inline]

Definition at line 359 of file kmlMotBase.h.

11.13.3.49 void CMotBase::setAccelerationLimit (short *acceleration*)

Set the acceleration limits.

11.13.3.50 void CMotBase::setPwmLimits (byte *maxppwm*, byte *maxnpwm*)

Set the PWM limits.

11.13.3.51 void CMotBase::setControllerParameters (byte *kSpeed*, byte *kPos*, byte *kI*)

Set the controller parameters.

11.13.3.52 void CMotBase::setCrashLimit (int *limit*)

Set the crash limit.

11.13.3.53 void CMotBase::setCrashLimitLinear (int *limit_lin*)

Set the crash limit linear.

11.13.3.54 void CMotBase::setSpeedCollisionLimit (int *limit*)

Set the collision limit.

11.13.3.55 void CMotBase::setPositionCollisionLimit (int *limit*)

Set the collision limit.

11.13.3.56 void CMotBase::getParameterOrLimit (int *subcommand*, byte * *R1*, byte * *R2*, byte * *R3*)

Get parameters or limits.

Parameters:

subcommand 255-249;245, see katana user manual chapter 8 firmware commands for details

R1 pointer to store first byte of answer

R2 pointer to store second byte of answer

R3 pointer to store third byte of answer

11.13.4 Friends And Related Function Documentation**11.13.4.1 friend class CKatBase** [friend]

Definition at line 221 of file kmlMotBase.h.

11.13.5 Member Data Documentation**11.13.5.1 TMotGNL CMotBase::gnl** [protected]

motor generals

Definition at line 225 of file kmlMotBase.h.

11.13.5.2 TMotAPS CMotBase::aps [protected]

actual position

Definition at line 226 of file kmlMotBase.h.

11.13.5.3 [TMotTPS CMotBase::tps](#) [protected]

target position

Definition at line 227 of file kmlMotBase.h.

11.13.5.4 [TMotSCP CMotBase::scp](#) [protected]

static controller parameters

Definition at line 228 of file kmlMotBase.h.

11.13.5.5 [TMotDYL CMotBase::dyl](#) [protected]

dynamic limits

Definition at line 229 of file kmlMotBase.h.

11.13.5.6 [TMotPVP CMotBase::pvp](#) [protected]

reading motor parameters

Definition at line 230 of file kmlMotBase.h.

11.13.5.7 [TMotSFW CMotBase::sfw](#) [protected]

slave firmware

Definition at line 231 of file kmlMotBase.h.

11.13.5.8 [TMotCLB CMotBase::_calibrationParameters](#) [protected]

calibration structure

Definition at line 232 of file kmlMotBase.h.

11.13.5.9 [TMotENL CMotBase::_encoderLimits](#) [protected]

motor limits in encoder values

Definition at line 233 of file kmlMotBase.h.

11.13.5.10 [TMotInit CMotBase::_initialParameters](#) [protected]

Definition at line 234 of file kmlMotBase.h.

11.13.5.11 [bool CMotBase::freedom](#) [protected]

if it is set, it will move on a parallel movement

Definition at line 235 of file kmlMotBase.h.

11.13.5.12 `bool CMotBase::nmp` [protected]

true if new motor parameters are implemented on the firmware

Definition at line 236 of file `kmlMotBase.h`.

11.13.5.13 `bool CMotBase::blocked` [protected]

true if the motor was blocked due to a crash of the robot

Definition at line 237 of file `kmlMotBase.h`.

11.13.5.14 `CCplBase* CMotBase::protocol` [protected]

protocol interface

Definition at line 264 of file `kmlMotBase.h`.

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

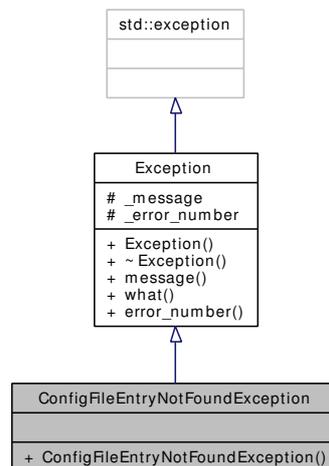
- `include/KNI/kmlMotBase.h`

11.14 ConfigFileEntryNotFoundException Class Reference

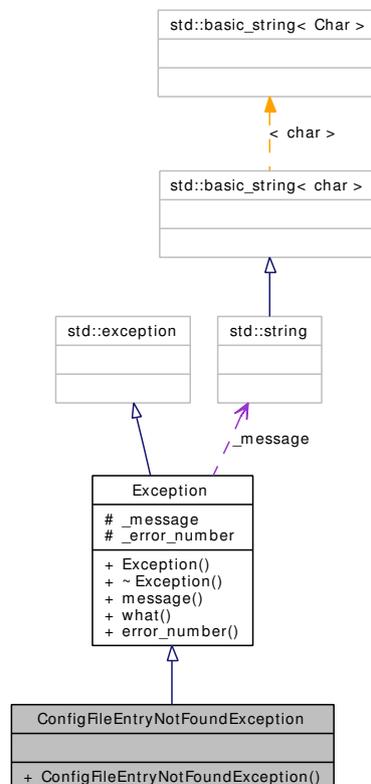
The requested entry could not be found.

```
#include <kmlFactories.h>
```

Inheritance diagram for ConfigFileEntryNotFoundException:



Collaboration diagram for ConfigFileEntryNotFoundException:



Public Member Functions

- [ConfigFileEntryNotFoundException](#) (const std::string &attribute) throw ()

11.14.1 Detailed Description

The requested entry could not be found.

Note:

error_number=-44

Definition at line 49 of file kmlFactories.h.

11.14.2 Constructor & Destructor Documentation

11.14.2.1 [ConfigFileEntryNotFoundException::ConfigFileEntryNotFoundException](#) (const std::string & *attribute*) throw () [inline]

Definition at line 51 of file kmlFactories.h.

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

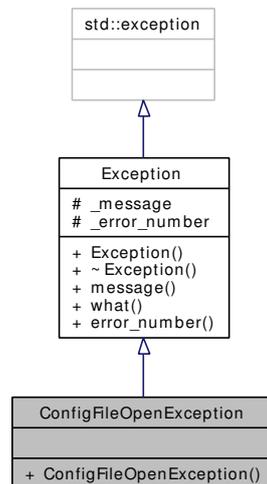
- [include/KNI/kmlFactories.h](#)

11.15 ConfigFileOpenException Class Reference

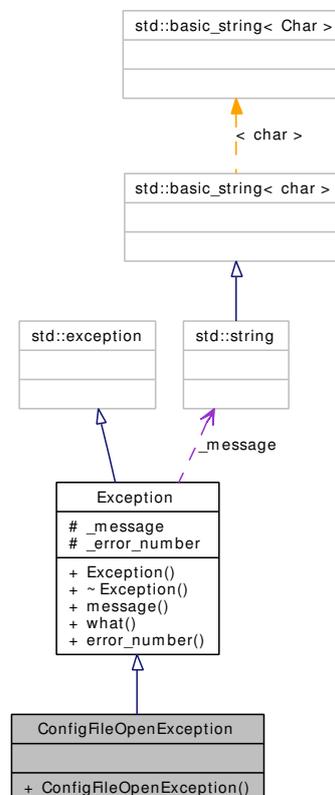
Accessing the given configuration file failed (may be: access denied or wrong path).

```
#include <kmlExt.h>
```

Inheritance diagram for ConfigFileOpenException:



Collaboration diagram for ConfigFileOpenException:



Public Member Functions

- [ConfigFileOpenException](#) (const std::string &port) throw ()

11.15.1 Detailed Description

Accessing the given configuration file failed (may be: access denied or wrong path).

Note:

error_number=-40

Definition at line 45 of file kmlExt.h.

11.15.2 Constructor & Destructor Documentation

11.15.2.1 [ConfigFileOpenException::ConfigFileOpenException](#) (const std::string &port) throw () [inline]

Definition at line 47 of file kmlExt.h.

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

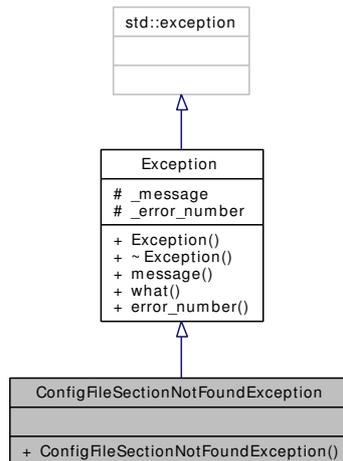
- [include/KNI/kmlExt.h](#)

11.16 ConfigFileSectionNotFoundException Class Reference

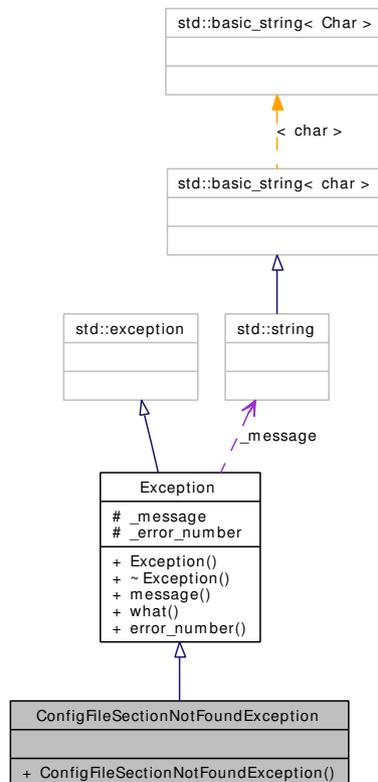
The requested section could not be found.

```
#include <kmlFactories.h>
```

Inheritance diagram for ConfigFileSectionNotFoundException:



Collaboration diagram for ConfigFileSectionNotFoundException:



Public Member Functions

- [ConfigFileSectionNotFoundException](#) (const std::string &attribute) throw ()

11.16.1 Detailed Description

The requested section could not be found.

Note:

error_number=-42

Definition at line 31 of file kmlFactories.h.

11.16.2 Constructor & Destructor Documentation

11.16.2.1 ConfigFileSectionNotFoundException::ConfigFileSectionNotFoundException (const std::string & *attribute*) throw () [inline]

Definition at line 33 of file kmlFactories.h.

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

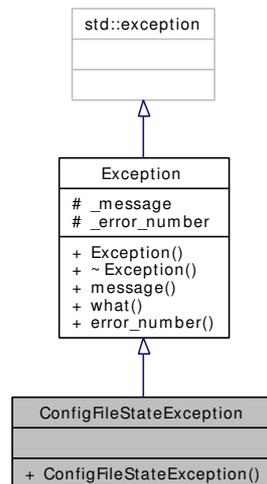
- [include/KNI/kmlFactories.h](#)

11.17 ConfigFileStateException Class Reference

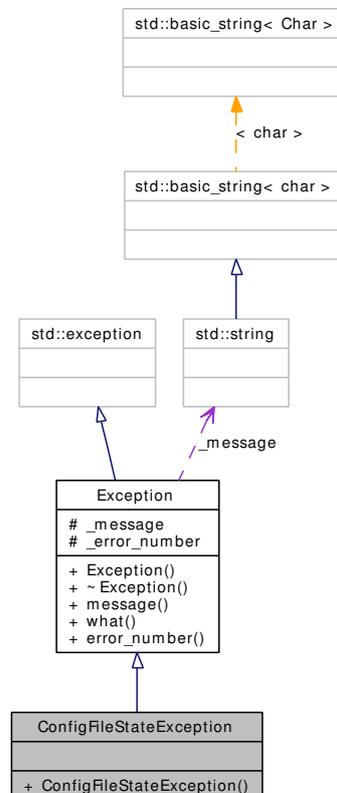
The state of the configuration file wasn't "good".

```
#include <kmlFactories.h>
```

Inheritance diagram for ConfigFileStateException:



Collaboration diagram for ConfigFileStateException:



Public Member Functions

- [ConfigFileStateException](#) () throw ()

11.17.1 Detailed Description

The state of the configuration file wasn't "good".

Note:

```
error_number=-41
```

Definition at line 22 of file [kmlFactories.h](#).

11.17.2 Constructor & Destructor Documentation

11.17.2.1 [ConfigFileStateException::ConfigFileStateException \(\) throw \(\)](#) [inline]

Definition at line 24 of file [kmlFactories.h](#).

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

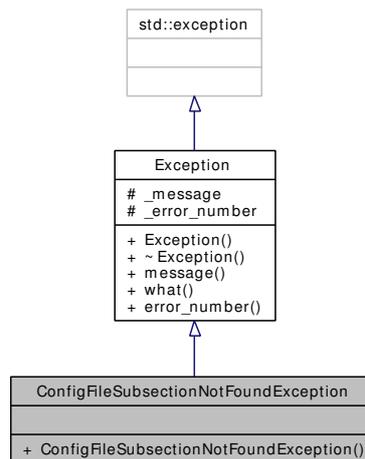
- [include/KNI/kmlFactories.h](#)

11.18 ConfigFileSubsectionNotFoundException Class Reference

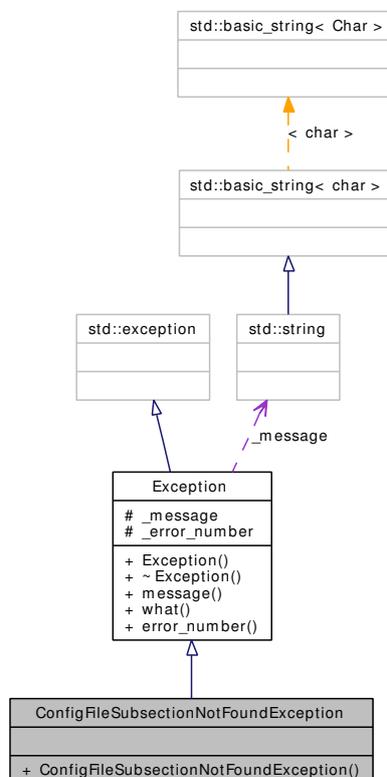
The requested subsection could not be found.

```
#include <kmlFactories.h>
```

Inheritance diagram for ConfigFileSubsectionNotFoundException:



Collaboration diagram for ConfigFileSubsectionNotFoundException:



Public Member Functions

- [ConfigFileSubsectionNotFoundException](#) (const std::string &attribute) throw ()

11.18.1 Detailed Description

The requested subsection could not be found.

Note:

error_number=-43

Definition at line 40 of file kmlFactories.h.

11.18.2 Constructor & Destructor Documentation

11.18.2.1 [ConfigFileSubsectionNotFoundException::ConfigFileSubsectionNotFoundException](#) (const std::string & *attribute*) throw () [inline]

Definition at line 42 of file kmlFactories.h.

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

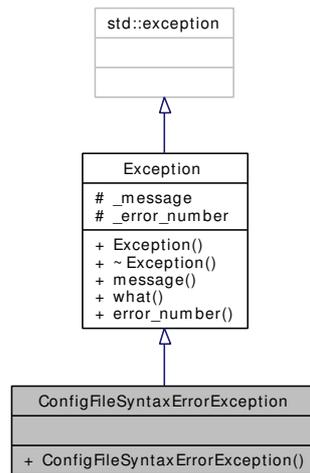
- [include/KNI/kmlFactories.h](#)

11.19 ConfigFileSyntaxErrorException Class Reference

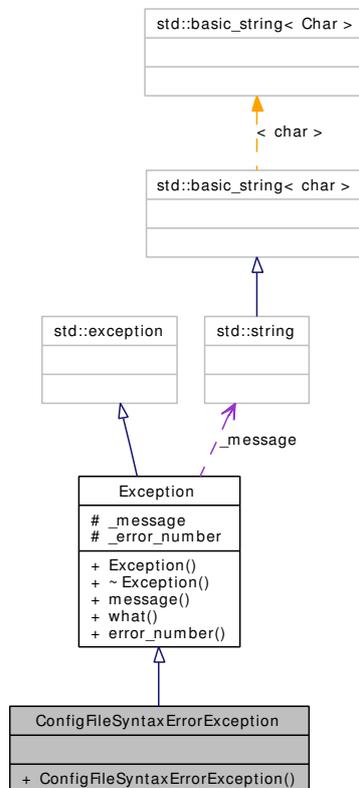
There was a syntax error in the configuration file.

```
#include <kmlFactories.h>
```

Inheritance diagram for ConfigFileSyntaxErrorException:



Collaboration diagram for ConfigFileSyntaxErrorException:



Public Member Functions

- [ConfigFileSyntaxErrorException](#) (const std::string &line) throw ()

11.19.1 Detailed Description

There was a syntax error in the configuration file.

Note:

error_number=-45

Definition at line 58 of file kmlFactories.h.

11.19.2 Constructor & Destructor Documentation

11.19.2.1 [ConfigFileSyntaxErrorException::ConfigFileSyntaxErrorException](#) (const std::string & *line*) throw () [inline]

Definition at line 60 of file kmlFactories.h.

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

- [include/KNI/kmlFactories.h](#)

11.20 Context Struct Reference

```
#include <exception.h>
```

Public Member Functions

- [Context](#) (const char *)

11.20.1 Detailed Description

Definition at line 75 of file exception.h.

11.20.2 Constructor & Destructor Documentation

11.20.2.1 Context::Context (const char *) [inline]

Definition at line 76 of file exception.h.

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

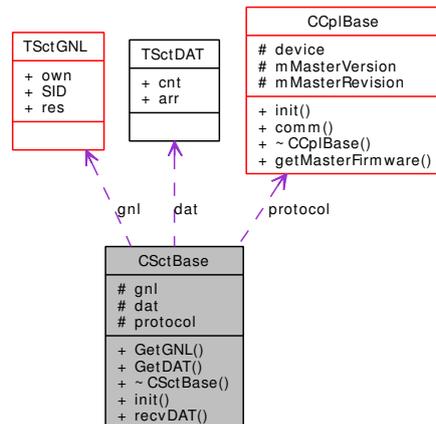
- [include/common/exception.h](#)

11.21 CSctBase Class Reference

Sensor Controller class.

```
#include <kmlSctBase.h>
```

Collaboration diagram for CSctBase:



Public Member Functions

- const [TSctGNL](#) * [GetGNL](#) ()
- const [TSctDAT](#) * [GetDAT](#) ()
- virtual [~CSctBase](#) ()
- bool [init](#) ([CKatBase](#) *_own, const [TSctDesc](#) _sctDesc, [CCplBase](#) *protocol)
- void [recvDAT](#) ()
receive data

Protected Attributes

- [TSctGNL](#) [gnl](#)
controller generals
- [TSctDAT](#) [dat](#)
sensor data
- [CCplBase](#) * [protocol](#)
protocol interface

Friends

- class [CKatBase](#)

11.21.1 Detailed Description

Sensor Controller class.

By using this class you can get access to the sensor data; to do so you should (after initialization) call 'recvDat()' to updated the internal 'TSctDAT dat' structure; after the updated you can read out the values by using the 'GetDAT()' function, which will return a constant pointer to the internal 'dat' structure.

Definition at line 72 of file kmlSctBase.h.

11.21.2 Constructor & Destructor Documentation

11.21.2.1 virtual CSctBase::~~CSctBase () [inline, virtual]

Definition at line 88 of file kmlSctBase.h.

11.21.3 Member Function Documentation

11.21.3.1 const TSctGNL* CSctBase::GetGNL () [inline]

Definition at line 81 of file kmlSctBase.h.

References CKatBase::gnl.

11.21.3.2 const TSctDAT* CSctBase::GetDAT () [inline]

Definition at line 82 of file kmlSctBase.h.

11.21.3.3 bool CSctBase::init (CKatBase * *_own*, const TSctDesc *_sctDesc*, CCplBase * *protocol*)

11.21.3.4 void CSctBase::recvDAT ()

receive data

11.21.4 Friends And Related Function Documentation

11.21.4.1 friend class CKatBase [friend]

Definition at line 74 of file kmlSctBase.h.

11.21.5 Member Data Documentation

11.21.5.1 TSctGNL CSctBase::gnl [protected]

controller generals

Definition at line 77 of file kmlSctBase.h.

11.21.5.2 TSctDAT CSctBase::dat [protected]

sensor data

Definition at line 78 of file kmlSctBase.h.

11.21.5.3 CCplBase* CSctBase::protocol [protected]

protocol interface

Definition at line 85 of file kmlSctBase.h.

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

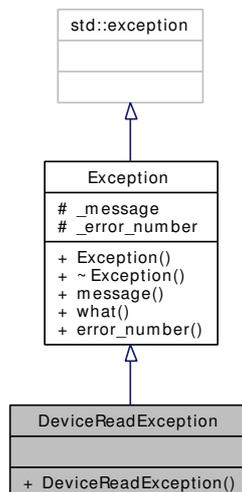
- [include/KNI/kmlSctBase.h](#)

11.22 DeviceReadException Class Reference

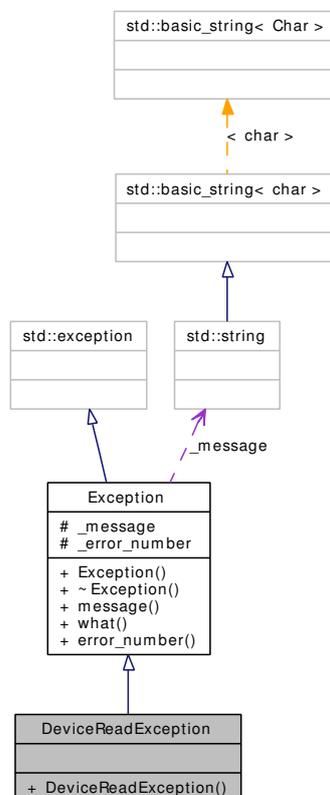
Reading from the serial communication device failed.

```
#include <cdlCOMExceptions.h>
```

Inheritance diagram for DeviceReadException:



Collaboration diagram for DeviceReadException:



Public Member Functions

- [DeviceReadException](#) (const std::string &port, const std::string os_msg) throw ()

11.22.1 Detailed Description

Reading from the serial communication device failed.

Note:

error_number=-13

Linux only: You get also the direct error message from the system

Definition at line 75 of file `cdlCOMExceptions.h`.

11.22.2 Constructor & Destructor Documentation

11.22.2.1 `DeviceReadException::DeviceReadException (const std::string &port, const std::string os_msg) throw ()` [inline]

Definition at line 77 of file `cdlCOMExceptions.h`.

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

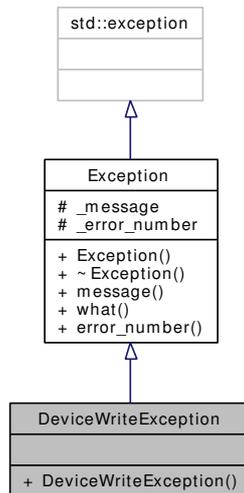
- [include/KNI/cdlCOMExceptions.h](#)

11.23 DeviceWriteException Class Reference

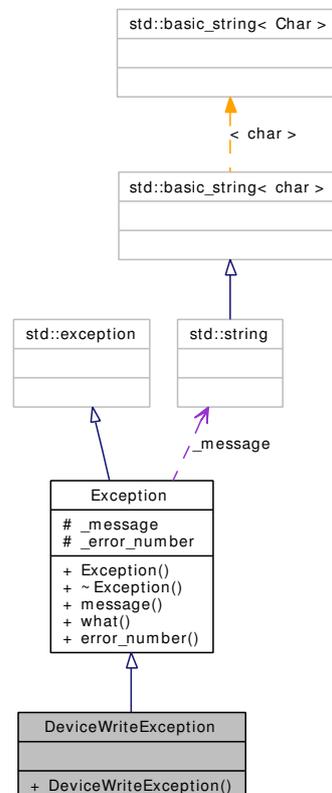
Writing to the serial communication device failed.

```
#include <cdlCOMExceptions.h>
```

Inheritance diagram for DeviceWriteException:



Collaboration diagram for DeviceWriteException:



Public Member Functions

- [DeviceWriteException](#) (const std::string &port, const std::string os_msg) throw ()

11.23.1 Detailed Description

Writing to the serial communication device failed.

Note:

error_number=-14

Linux only: You get also the direct error message from the system

Definition at line 85 of file `cdlCOMExceptions.h`.

11.23.2 Constructor & Destructor Documentation

11.23.2.1 DeviceWriteException::DeviceWriteException (const std::string & port, const std::string os_msg) throw () [inline]

Definition at line 87 of file `cdlCOMExceptions.h`.

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

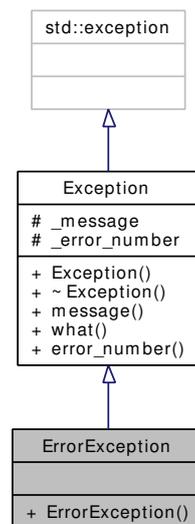
- [include/KNI/cdlCOMExceptions.h](#)

11.24 RecognitionException Class Reference

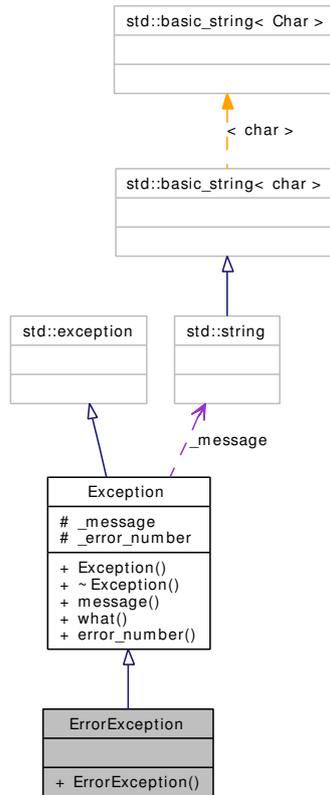
The Katana returned an error string.

```
#include <cdlCOMExceptions.h>
```

Inheritance diagram for RecognitionException:



Collaboration diagram for RecognitionException:



Public Member Functions

- [ErrorException](#) (const std::string &error) throw ()

11.24.1 Detailed Description

The Katana returned an error string.

Note:

error_number=-16

Definition at line 121 of file cdlCOMExceptions.h.

11.24.2 Constructor & Destructor Documentation

11.24.2.1 ErrorException::ErrorException (const std::string & error) throw () [inline]

Definition at line 123 of file cdlCOMExceptions.h.

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

- [include/KNI/cdlCOMExceptions.h](#)

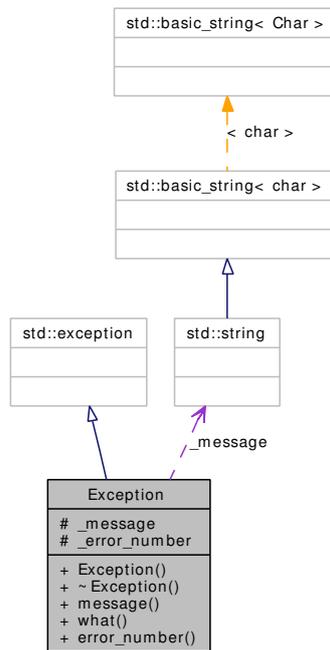
11.25 Exception Class Reference

```
#include <exception.h>
```

Inheritance diagram for Exception:



Collaboration diagram for Exception:



Public Member Functions

- [Exception](#) (const std::string &message, const int error_number) throw ()
- virtual [~Exception](#) () throw ()
- std::string [message](#) () const throw ()
- const char * [what](#) () const throw ()
- const int [error_number](#) () const throw ()

Protected Attributes

- const std::string [_message](#)
- const int [_error_number](#)

11.25.1 Detailed Description

Definition at line 79 of file exception.h.

11.25.2 Constructor & Destructor Documentation

11.25.2.1 `Exception::Exception (const std::string & message, const int error_number) throw ()` [inline]

Definition at line 85 of file exception.h.

11.25.2.2 `virtual Exception::~Exception () throw ()` [inline, virtual]

Definition at line 90 of file exception.h.

11.25.3 Member Function Documentation

11.25.3.1 `std::string Exception::message () const throw ()` [inline]

Definition at line 93 of file exception.h.

11.25.3.2 `const char* Exception::what () const throw ()` [inline]

Definition at line 96 of file exception.h.

11.25.3.3 `const int Exception::error_number () const throw ()` [inline]

Definition at line 100 of file exception.h.

11.25.4 Member Data Documentation

11.25.4.1 `const std::string Exception::_message` [protected]

Definition at line 81 of file exception.h.

11.25.4.2 `const int Exception::_error_number` [protected]

Definition at line 82 of file exception.h.

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

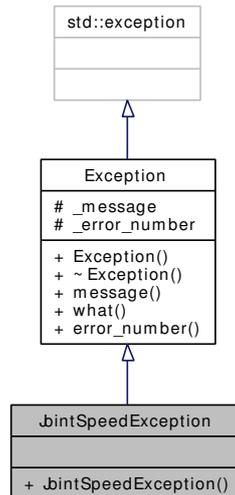
- [include/common/exception.h](#)

11.26 JointSpeedException Class Reference

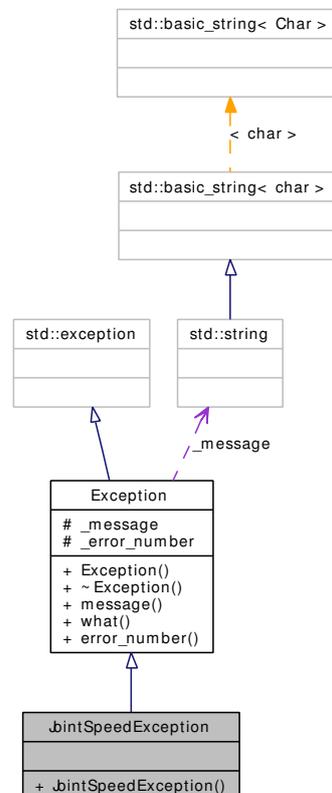
Joint speed too high.

```
#include <lmBase.h>
```

Inheritance diagram for JointSpeedException:



Collaboration diagram for JointSpeedException:



Public Member Functions

- [JointSpeedException](#) () throw ()

11.26.1 Detailed Description

Joint speed too high.

Note:

error_number = -70

Definition at line 129 of file lmBase.h.

11.26.2 Constructor & Destructor Documentation

11.26.2.1 [JointSpeedException::JointSpeedException](#) () throw () [inline]

Definition at line 131 of file lmBase.h.

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

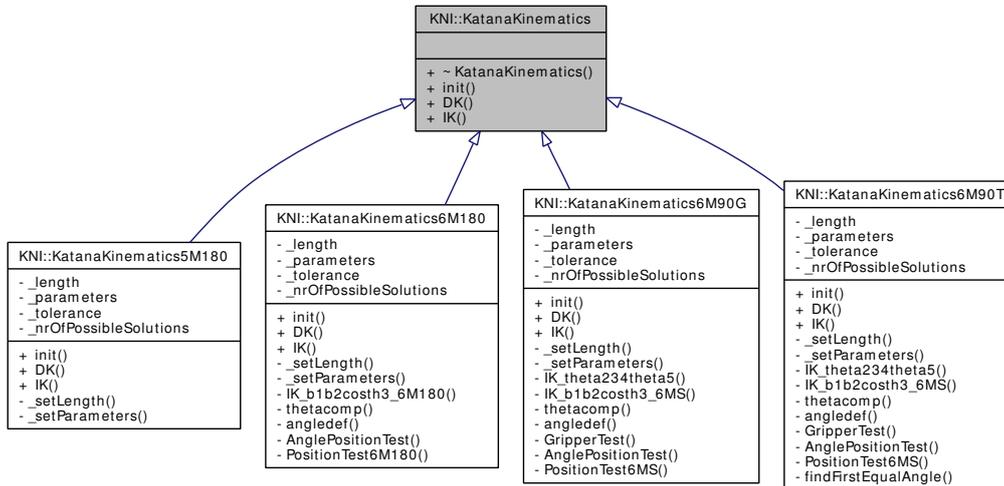
- [include/KNI_LM/lmBase.h](#)

11.27 KNI::KatanaKinematics Class Reference

The base class for all kinematic implementations.

```
#include <KatanaKinematics.h>
```

Inheritance diagram for KNI::KatanaKinematics:



Public Types

- typedef std::vector< [KinematicParameters](#) > [parameter_container](#)
- typedef std::vector< double > [angles](#)
Being used to store angles (in radian).
- typedef std::vector< double > [coordinates](#)
To store coordinates.
- typedef std::vector< double > [metrics](#)
To store metrics, 'aka' the length's of the different segments of the robot.
- typedef std::vector< int > [encoders](#)
To store encoders.

Public Member Functions

- virtual [~KatanaKinematics](#) ()
- virtual void [init](#) ([metrics](#) const &length, [parameter_container](#) const ¶meters)=0
Initialize the parameters for the calculations.
- virtual void [DK](#) ([coordinates](#) &solution, [encoders](#) const ¤t_encoders) const=0
Direct Kinematic.

- virtual void [IK](#) (encoders::iterator solution, [coordinates](#) const &pose, [encoders](#) const &cur_angles) const=0

Inverse Kinematic.

11.27.1 Detailed Description

The base class for all kinematic implementations.

Definition at line 63 of file KatanaKinematics.h.

11.27.2 Member Typedef Documentation

11.27.2.1 `typedef std::vector<KinematicParameters> KNI::KatanaKinematics::parameter_container`

Definition at line 67 of file KatanaKinematics.h.

11.27.2.2 `typedef std::vector<double> KNI::KatanaKinematics::angles`

Being used to store angles (in radian).

Definition at line 71 of file KatanaKinematics.h.

11.27.2.3 `typedef std::vector<double> KNI::KatanaKinematics::coordinates`

To store coordinates.

Definition at line 74 of file KatanaKinematics.h.

11.27.2.4 `typedef std::vector<double> KNI::KatanaKinematics::metrics`

To store metrics, 'aka' the length's of the different segments of the robot.

Definition at line 77 of file KatanaKinematics.h.

11.27.2.5 `typedef std::vector<int> KNI::KatanaKinematics::encoders`

To store encoders.

Definition at line 80 of file KatanaKinematics.h.

11.27.3 Constructor & Destructor Documentation

11.27.3.1 `virtual KNI::KatanaKinematics::~KatanaKinematics () [inline, virtual]`

Definition at line 65 of file KatanaKinematics.h.

11.27.4 Member Function Documentation

11.27.4.1 virtual void KNI::KatanaKinematics::init ([metrics](#) const & [length](#), [parameter_container](#) const & [parameters](#)) [pure virtual]

Initialize the parameters for the calculations.

This is needed to validate the calculated angles and to choose an appropriate solution You have to provide 5 or 6 length's and parameters, depending on you robot type

11.27.4.2 virtual void KNI::KatanaKinematics::DK ([coordinates](#) & [solution](#), [encoders](#) const & [current_encoders](#)) const [pure virtual]

Direct Kinematic.

Calculates the actual position in cartesian coordinates using the given encoders

Parameters:

[solution](#) This is where the algorithm will store the solution to (in cartesian coordinates)

[current_encoders](#) The encoder values which are being used for the calculation

Note:

strong guarantee provided

Implemented in [KNI::KatanaKinematics5M180](#), [KNI::KatanaKinematics6M180](#), [KNI::KatanaKinematics6M90G](#), and [KNI::KatanaKinematics6M90T](#).

11.27.4.3 virtual void KNI::KatanaKinematics::IK ([encoders::iterator](#) [solution](#), [coordinates](#) const & [pose](#), [encoders](#) const & [cur_angles](#)) const [pure virtual]

Inverse Kinematic.

Calculates one set of encoders (=one solution) for the given cartesian coordinates. You also have to provide the current encoders to allow the algorithm to choose between different valid solutions.

Parameters:

[solution](#) This is where the algorithm will store the solution to (in encoders)

[pose](#) The target position in cartesian coordinates plus the euler angles for the direction of the gripper

[cur_angles](#) The current angles (in encoders) of the robot

Note:

strong guarantee provided

Implemented in [KNI::KatanaKinematics5M180](#), [KNI::KatanaKinematics6M180](#), [KNI::KatanaKinematics6M90G](#), and [KNI::KatanaKinematics6M90T](#).

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

- [include/KNI_InvKin/KatanaKinematics.h](#)

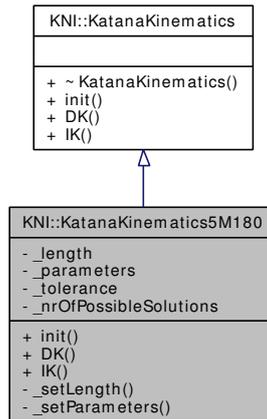
11.28 KNI::KatanaKinematics5M180 Class Reference

Author:

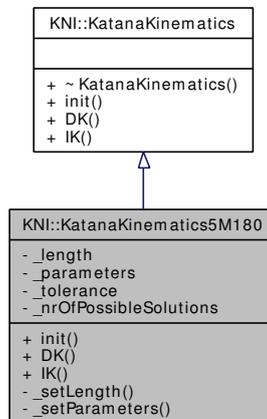
Tiziano Mueller <tiziano.mueller@neuronics.ch>

```
#include <KatanaKinematics5M180.h>
```

Inheritance diagram for KNI::KatanaKinematics5M180:



Collaboration diagram for KNI::KatanaKinematics5M180:



Public Member Functions

- void **init** ([metrics](#) const &length, [parameter_container](#) const ¶meters)
- void **DK** ([coordinates](#) &solution, [encoders](#) const ¤t_encoders) const
Direct Kinematic.
- void **IK** ([encoders::iterator](#) solution, [coordinates](#) const &pose, [encoders](#) const &cur_angles) const
Inverse Kinematic.

Private Types

- typedef std::vector< [angles_calc](#) > [angles_container](#)

Private Member Functions

- void [_setLength](#) ([metrics](#) const &length)
- void [_setParameters](#) ([parameter_container](#) const ¶meters)

Private Attributes

- [metrics_length](#)
- [parameter_container_parameters](#)

Static Private Attributes

- static const double [_tolerance](#)
- static const int [_nrOfPossibleSolutions](#)

Classes

- struct [angles_calc](#)
- struct [position](#)

11.28.1 Detailed Description

Author:

Tiziano Mueller <tiziano.mueller@neuronics.ch>

Author:

Christoph Voser <christoph.voser@neuronics.ch>

Definition at line 39 of file KatanaKinematics5M180.h.

11.28.2 Member Typedef Documentation

11.28.2.1 typedef std::vector<[angles_calc](#)> [KNI::KatanaKinematics5M180::angles_container](#) [private]

Definition at line 70 of file KatanaKinematics5M180.h.

11.28.3 Member Function Documentation

11.28.3.1 void KNI::KatanaKinematics5M180::init (**metrics** const & *length*, **parameter_container** const & *parameters*)

11.28.3.2 void KNI::KatanaKinematics5M180::DK (**coordinates** & *solution*, **encoders** const & *current_encoders*) const [virtual]

Direct Kinematic.

Calculates the actual position in cartesian coordinates using the given encoders

Parameters:

solution This is where the algorithm will store the solution to (in cartesian coordinates)

current_encoders The encoder values which are being used for the calculation

Note:

strong guarantee provided

Implements [KNI::KatanaKinematics](#).

11.28.3.3 void KNI::KatanaKinematics5M180::IK (**encoders::iterator** *solution*, **coordinates** const & *pose*, **encoders** const & *cur_angles*) const [virtual]

Inverse Kinematic.

Calculates one set of encoders (=one solution) for the given cartesian coordinates. You also have to provide the current encoders to allow the algorithm to choose between different valid solutions.

Parameters:

solution This is where the algorithm will store the solution to (in encoders)

pose The target position in cartesian coordinates plus the euler angles for the direction of the gripper

cur_angles The current angles (in encoders) of the robot

Note:

strong guarantee provided

Implements [KNI::KatanaKinematics](#).

11.28.3.4 void KNI::KatanaKinematics5M180::_setLength (**metrics** const & *length*) [inline, private]

Definition at line 78 of file KatanaKinematics5M180.h.

11.28.3.5 void KNI::KatanaKinematics5M180::_setParameters (**parameter_container** const & *parameters*) [inline, private]

Definition at line 79 of file KatanaKinematics5M180.h.

11.28.4 Member Data Documentation

11.28.4.1 `metrics KNI::KatanaKinematics5M180::_length` [private]

Definition at line 72 of file `KatanaKinematics5M180.h`.

11.28.4.2 `parameter_container KNI::KatanaKinematics5M180::_parameters` [private]

Definition at line 73 of file `KatanaKinematics5M180.h`.

11.28.4.3 `const double KNI::KatanaKinematics5M180::_tolerance` [static, private]

Definition at line 75 of file `KatanaKinematics5M180.h`.

11.28.4.4 `const int KNI::KatanaKinematics5M180::_nrOfPossibleSolutions` [static, private]

Definition at line 76 of file `KatanaKinematics5M180.h`.

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

- `include/KNI_InvKin/KatanaKinematics5M180.h`

11.29 KNI::KatanaKinematics5M180::angles_calc Struct Reference

Public Attributes

- double [theta1](#)
- double [theta2](#)
- double [theta3](#)
- double [theta4](#)
- double [theta5](#)
- double [theta234](#)
- double [b1](#)
- double [b2](#)
- double [costh3](#)

11.29.1 Detailed Description

Definition at line 58 of file KatanaKinematics5M180.h.

11.29.2 Member Data Documentation

11.29.2.1 double [KNI::KatanaKinematics5M180::angles_calc::theta1](#)

Definition at line 59 of file KatanaKinematics5M180.h.

11.29.2.2 double [KNI::KatanaKinematics5M180::angles_calc::theta2](#)

Definition at line 60 of file KatanaKinematics5M180.h.

11.29.2.3 double [KNI::KatanaKinematics5M180::angles_calc::theta3](#)

Definition at line 61 of file KatanaKinematics5M180.h.

11.29.2.4 double [KNI::KatanaKinematics5M180::angles_calc::theta4](#)

Definition at line 62 of file KatanaKinematics5M180.h.

11.29.2.5 double [KNI::KatanaKinematics5M180::angles_calc::theta5](#)

Definition at line 63 of file KatanaKinematics5M180.h.

11.29.2.6 double [KNI::KatanaKinematics5M180::angles_calc::theta234](#)

Definition at line 64 of file KatanaKinematics5M180.h.

11.29.2.7 double KNI::KatanaKinematics5M180::angles_calc::b1

Definition at line 65 of file KatanaKinematics5M180.h.

11.29.2.8 double KNI::KatanaKinematics5M180::angles_calc::b2

Definition at line 66 of file KatanaKinematics5M180.h.

11.29.2.9 double KNI::KatanaKinematics5M180::angles_calc::costh3

Definition at line 67 of file KatanaKinematics5M180.h.

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

- include/KNI_InvKin/KatanaKinematics5M180.h

11.30 KNI::KatanaKinematics5M180::position Struct Reference

Public Attributes

- double [x](#)
- double [y](#)
- double [z](#)

11.30.1 Detailed Description

Definition at line 52 of file [KatanaKinematics5M180.h](#).

11.30.2 Member Data Documentation

11.30.2.1 double [KNI::KatanaKinematics5M180::position::x](#)

Definition at line 53 of file [KatanaKinematics5M180.h](#).

11.30.2.2 double [KNI::KatanaKinematics5M180::position::y](#)

Definition at line 54 of file [KatanaKinematics5M180.h](#).

11.30.2.3 double [KNI::KatanaKinematics5M180::position::z](#)

Definition at line 55 of file [KatanaKinematics5M180.h](#).

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

- [include/KNI_InvKin/KatanaKinematics5M180.h](#)

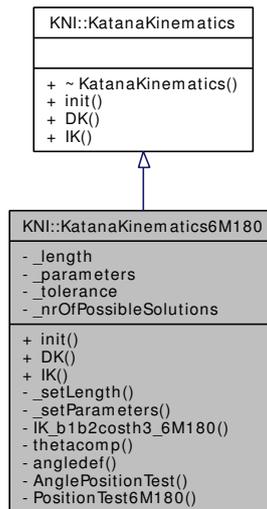
11.31 KNI::KatanaKinematics6M180 Class Reference

Author:

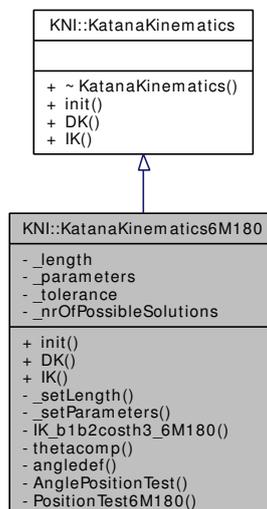
Tiziano Mueller <tiziano.mueller@neuronics.ch>

```
#include <KatanaKinematics6M180.h>
```

Inheritance diagram for KNI::KatanaKinematics6M180:



Collaboration diagram for KNI::KatanaKinematics6M180:



Public Member Functions

- void [init](#) ([metrics](#) const &length, [parameter_container](#) const ¶meters)
- void [DK](#) ([coordinates](#) &solution, [encoders](#) const ¤t_encoders) const

Direct Kinematic.

- void **IK** (encoders::iterator solution, [coordinates](#) const &pose, [encoders](#) const &cur_angles) const

Inverse Kinematic.

Private Types

- typedef std::vector< [angles_calc](#) > [angles_container](#)

Private Member Functions

- void [_setLength](#) ([metrics](#) const &length)
- void [_setParameter](#)s ([parameter_container](#) const ¶meters)
- void [IK_b1b2cosh3_6M180](#) ([angles_calc](#) &a, const [position](#) &p) const
- void [thetacom](#)p ([angles_calc](#) &a, const [position](#) &p_m) const
- bool [angledef](#) ([angles_calc](#) &a) const
- bool [AnglePositionTest](#) (const [angles_calc](#) &a) const
- bool [PositionTest6M180](#) (const [angles_calc](#) &a, const [position](#) &p) const

Private Attributes

- [metrics_length](#)
- [parameter_container_parameters](#)

Static Private Attributes

- static const double [_tolerance](#)
- static const int [_nrOfPossibleSolutions](#)

Classes

- struct [angles_calc](#)
- struct [position](#)

11.31.1 Detailed Description

Author:

Tiziano Mueller <tiziano.mueller@neuronics.ch>

Author:

Christoph Voser <christoph.voser@neuronics.ch>

Definition at line 40 of file [KatanaKinematics6M180.h](#).

11.31.2 Member Typedef Documentation

11.31.2.1 `typedef std::vector<angles_calc> KNI::KatanaKinematics6M180::angles_container`
[private]

Definition at line 71 of file KatanaKinematics6M180.h.

11.31.3 Member Function Documentation

11.31.3.1 `void KNI::KatanaKinematics6M180::init (metrics const & length, parameter_container const & parameters)`

11.31.3.2 `void KNI::KatanaKinematics6M180::DK (coordinates & solution, encoders const & current_encoders) const` [virtual]

Direct Kinematic.

Calculates the actual position in cartesian coordinates using the given encoders

Parameters:

solution This is where the algorithm will store the solution to (in cartesian coordinates)

current_encoders The encoder values which are being used for the calculation

Note:

strong guarantee provided

Implements [KNI::KatanaKinematics](#).

11.31.3.3 `void KNI::KatanaKinematics6M180::IK (encoders::iterator solution, coordinates const & pose, encoders const & cur_angles) const` [virtual]

Inverse Kinematic.

Calculates one set of encoders (=one solution) for the given cartesian coordinates. You also have to provide the current encoders to allow the algorithm to choose between different valid solutions.

Parameters:

solution This is where the algorithm will store the solution to (in encoders)

pose The target position in cartesian coordinates plus the euler angles for the direction of the gripper

cur_angles The current angles (in encoders) of the robot

Note:

strong guarantee provided

Implements [KNI::KatanaKinematics](#).

11.31.3.4 `void KNI::KatanaKinematics6M180::_setLength (metrics const & length)` [inline, private]

Definition at line 79 of file KatanaKinematics6M180.h.

11.31.3.5 void KNI::KatanaKinematics6M180::_setParameters ([parameter_container](#) const & [parameters](#)) [inline, private]

Definition at line 80 of file KatanaKinematics6M180.h.

11.31.3.6 void KNI::KatanaKinematics6M180::IK_b1b2costh3_6M180 ([angles_calc](#) & *a*, const [position](#) & *p*) const [private]

11.31.3.7 void KNI::KatanaKinematics6M180::thetacomp ([angles_calc](#) & *a*, const [position](#) & *p_m*) const [private]

11.31.3.8 bool KNI::KatanaKinematics6M180::angledef ([angles_calc](#) & *a*) const [private]

11.31.3.9 bool KNI::KatanaKinematics6M180::AnglePositionTest (const [angles_calc](#) & *a*) const [private]

11.31.3.10 bool KNI::KatanaKinematics6M180::PositionTest6M180 (const [angles_calc](#) & *a*, const [position](#) & *p*) const [private]

11.31.4 Member Data Documentation

11.31.4.1 [metrics](#) KNI::KatanaKinematics6M180::_length [private]

Definition at line 73 of file KatanaKinematics6M180.h.

11.31.4.2 [parameter_container](#) KNI::KatanaKinematics6M180::_parameters [private]

Definition at line 74 of file KatanaKinematics6M180.h.

11.31.4.3 const double KNI::KatanaKinematics6M180::_tolerance [static, private]

Definition at line 76 of file KatanaKinematics6M180.h.

11.31.4.4 const int KNI::KatanaKinematics6M180::_nrOfPossibleSolutions [static, private]

Definition at line 77 of file KatanaKinematics6M180.h.

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

- include/KNI_InvKin/KatanaKinematics6M180.h

11.32 KNI::KatanaKinematics6M180::angles_calc Struct Reference

Public Attributes

- double [theta1](#)
- double [theta2](#)
- double [theta3](#)
- double [theta4](#)
- double [theta5](#)
- double [theta234](#)
- double [b1](#)
- double [b2](#)
- double [costh3](#)

11.32.1 Detailed Description

Definition at line 59 of file KatanaKinematics6M180.h.

11.32.2 Member Data Documentation

11.32.2.1 double [KNI::KatanaKinematics6M180::angles_calc::theta1](#)

Definition at line 60 of file KatanaKinematics6M180.h.

11.32.2.2 double [KNI::KatanaKinematics6M180::angles_calc::theta2](#)

Definition at line 61 of file KatanaKinematics6M180.h.

11.32.2.3 double [KNI::KatanaKinematics6M180::angles_calc::theta3](#)

Definition at line 62 of file KatanaKinematics6M180.h.

11.32.2.4 double [KNI::KatanaKinematics6M180::angles_calc::theta4](#)

Definition at line 63 of file KatanaKinematics6M180.h.

11.32.2.5 double [KNI::KatanaKinematics6M180::angles_calc::theta5](#)

Definition at line 64 of file KatanaKinematics6M180.h.

11.32.2.6 double [KNI::KatanaKinematics6M180::angles_calc::theta234](#)

Definition at line 65 of file KatanaKinematics6M180.h.

11.32.2.7 double [KNI::KatanaKinematics6M180::angles_calc::b1](#)

Definition at line 66 of file KatanaKinematics6M180.h.

11.32.2.8 double [KNI::KatanaKinematics6M180::angles_calc::b2](#)

Definition at line 67 of file KatanaKinematics6M180.h.

11.32.2.9 double [KNI::KatanaKinematics6M180::angles_calc::costh3](#)

Definition at line 68 of file KatanaKinematics6M180.h.

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

- include/KNI_InvKin/KatanaKinematics6M180.h

11.33 KNI::KatanaKinematics6M180::position Struct Reference

Public Attributes

- double [x](#)
- double [y](#)
- double [z](#)

11.33.1 Detailed Description

Definition at line 53 of file [KatanaKinematics6M180.h](#).

11.33.2 Member Data Documentation

11.33.2.1 double [KNI::KatanaKinematics6M180::position::x](#)

Definition at line 54 of file [KatanaKinematics6M180.h](#).

11.33.2.2 double [KNI::KatanaKinematics6M180::position::y](#)

Definition at line 55 of file [KatanaKinematics6M180.h](#).

11.33.2.3 double [KNI::KatanaKinematics6M180::position::z](#)

Definition at line 56 of file [KatanaKinematics6M180.h](#).

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

- [include/KNI_InvKin/KatanaKinematics6M180.h](#)

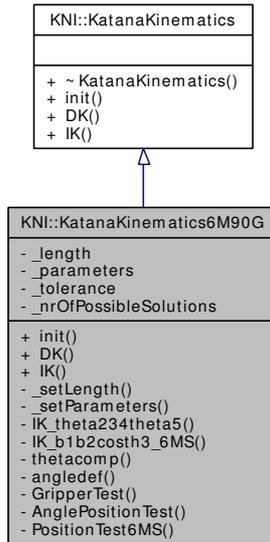
11.34 KNI::KatanaKinematics6M90G Class Reference

Author:

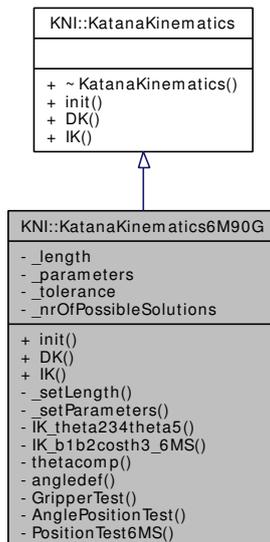
Tiziano Mueller <tiziano.mueller@neuronics.ch>

```
#include <KatanaKinematics6M90G.h>
```

Inheritance diagram for KNI::KatanaKinematics6M90G:



Collaboration diagram for KNI::KatanaKinematics6M90G:



Public Member Functions

- void `init` (`metrics` const &`length`, `parameter_container` const &`parameters`)

- void [DK](#) ([coordinates](#) &solution, [encoders](#) const ¤t_encoders) const
Direct Kinematic.
- void [IK](#) (encoders::iterator solution, [coordinates](#) const &pose, [encoders](#) const &cur_angles) const
Inverse Kinematic.

Private Types

- typedef std::vector< [angles_calc](#) > [angles_container](#)

Private Member Functions

- void [_setLength](#) ([metrics](#) const &length)
- void [_setParameters](#) ([parameter_container](#) const ¶meters)
- void [IK_theta234theta5](#) ([angles_calc](#) &angle, const [position](#) &p_gr) const
- void [IK_b1b2costh3_6MS](#) ([angles_calc](#) &a, const [position](#) &p) const
- void [thetacomp](#) ([angles_calc](#) &a, const [position](#) &p_m) const
- bool [angledef](#) ([angles_calc](#) &a) const
- bool [GripperTest](#) (const [position](#) &p_gr, const [angles_calc](#) &angle) const
- bool [AnglePositionTest](#) (const [angles_calc](#) &a) const
- bool [PositionTest6MS](#) (const [angles_calc](#) &a, const [position](#) &p) const

Private Attributes

- [metrics_length](#)
- [parameter_container_parameters](#)

Static Private Attributes

- static const double [_tolerance](#)
- static const int [_nrOfPossibleSolutions](#)

Classes

- struct [angles_calc](#)
- struct [position](#)

11.34.1 Detailed Description

Author:

Tiziano Mueller <tiziano.mueller@neuronics.ch>

Author:

Christoph Voser <christoph.voser@neuronics.ch>

Definition at line 39 of file [KatanaKinematics6M90G.h](#).

11.34.2 Member Typedef Documentation

11.34.2.1 `typedef std::vector<angles_calc> KNI::KatanaKinematics6M90G::angles_container`
[private]

Definition at line 70 of file KatanaKinematics6M90G.h.

11.34.3 Member Function Documentation

11.34.3.1 `void KNI::KatanaKinematics6M90G::init (metrics const & length, parameter_container const & parameters)`

11.34.3.2 `void KNI::KatanaKinematics6M90G::DK (coordinates & solution, encoders const & current_encoders) const` [virtual]

Direct Kinematic.

Calculates the actual position in cartesian coordinates using the given encoders

Parameters:

solution This is where the algorithm will store the solution to (in cartesian coordinates)

current_encoders The encoder values which are being used for the calculation

Note:

strong guarantee provided

Implements [KNI::KatanaKinematics](#).

11.34.3.3 `void KNI::KatanaKinematics6M90G::IK (encoders::iterator solution, coordinates const & pose, encoders const & cur_angles) const` [virtual]

Inverse Kinematic.

Calculates one set of encoders (=one solution) for the given cartesian coordinates. You also have to provide the current encoders to allow the algorithm to choose between different valid solutions.

Parameters:

solution This is where the algorithm will store the solution to (in encoders)

pose The target position in cartesian coordinates plus the euler angles for the direction of the gripper

cur_angles The current angles (in encoders) of the robot

Note:

strong guarantee provided

Implements [KNI::KatanaKinematics](#).

11.34.3.4 `void KNI::KatanaKinematics6M90G::_setLength (metrics const & length)` [inline, private]

Definition at line 78 of file KatanaKinematics6M90G.h.

11.34.3.5 void KNI::KatanaKinematics6M90G::_setParameters ([parameter_container](#) const & [parameters](#)) [[inline](#), [private](#)]

Definition at line 79 of file KatanaKinematics6M90G.h.

11.34.3.6 void KNI::KatanaKinematics6M90G::IK_theta234theta5 ([angles_calc](#) & [angle](#), const [position](#) & [p_gr](#)) const [[private](#)]

11.34.3.7 void KNI::KatanaKinematics6M90G::IK_b1b2costh3_6MS ([angles_calc](#) & [a](#), const [position](#) & [p](#)) const [[private](#)]

11.34.3.8 void KNI::KatanaKinematics6M90G::thetacomp ([angles_calc](#) & [a](#), const [position](#) & [p_m](#)) const [[private](#)]

11.34.3.9 bool KNI::KatanaKinematics6M90G::angledef ([angles_calc](#) & [a](#)) const [[private](#)]

11.34.3.10 bool KNI::KatanaKinematics6M90G::GripperTest (const [position](#) & [p_gr](#), const [angles_calc](#) & [angle](#)) const [[private](#)]

11.34.3.11 bool KNI::KatanaKinematics6M90G::AnglePositionTest (const [angles_calc](#) & [a](#)) const [[private](#)]

11.34.3.12 bool KNI::KatanaKinematics6M90G::PositionTest6MS (const [angles_calc](#) & [a](#), const [position](#) & [p](#)) const [[private](#)]

11.34.4 Member Data Documentation

11.34.4.1 [metrics](#) KNI::KatanaKinematics6M90G::_length [[private](#)]

Definition at line 72 of file KatanaKinematics6M90G.h.

11.34.4.2 [parameter_container](#) KNI::KatanaKinematics6M90G::_parameters [[private](#)]

Definition at line 73 of file KatanaKinematics6M90G.h.

11.34.4.3 const double KNI::KatanaKinematics6M90G::_tolerance [[static](#), [private](#)]

Definition at line 75 of file KatanaKinematics6M90G.h.

11.34.4.4 const int KNI::KatanaKinematics6M90G::_nrOfPossibleSolutions [[static](#), [private](#)]

Definition at line 76 of file KatanaKinematics6M90G.h.

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

- [include/KNI_InvKin/KatanaKinematics6M90G.h](#)

11.35 KNI::KatanaKinematics6M90G::angles_calc Struct Reference

Public Attributes

- double [theta1](#)
- double [theta2](#)
- double [theta3](#)
- double [theta4](#)
- double [theta5](#)
- double [theta234](#)
- double [b1](#)
- double [b2](#)
- double [costh3](#)

11.35.1 Detailed Description

Definition at line 58 of file KatanaKinematics6M90G.h.

11.35.2 Member Data Documentation

11.35.2.1 double [KNI::KatanaKinematics6M90G::angles_calc::theta1](#)

Definition at line 59 of file KatanaKinematics6M90G.h.

11.35.2.2 double [KNI::KatanaKinematics6M90G::angles_calc::theta2](#)

Definition at line 60 of file KatanaKinematics6M90G.h.

11.35.2.3 double [KNI::KatanaKinematics6M90G::angles_calc::theta3](#)

Definition at line 61 of file KatanaKinematics6M90G.h.

11.35.2.4 double [KNI::KatanaKinematics6M90G::angles_calc::theta4](#)

Definition at line 62 of file KatanaKinematics6M90G.h.

11.35.2.5 double [KNI::KatanaKinematics6M90G::angles_calc::theta5](#)

Definition at line 63 of file KatanaKinematics6M90G.h.

11.35.2.6 double [KNI::KatanaKinematics6M90G::angles_calc::theta234](#)

Definition at line 64 of file KatanaKinematics6M90G.h.

11.35.2.7 double [KNI::KatanaKinematics6M90G::angles_calc::b1](#)

Definition at line 65 of file [KatanaKinematics6M90G.h](#).

11.35.2.8 double [KNI::KatanaKinematics6M90G::angles_calc::b2](#)

Definition at line 66 of file [KatanaKinematics6M90G.h](#).

11.35.2.9 double [KNI::KatanaKinematics6M90G::angles_calc::costh3](#)

Definition at line 67 of file [KatanaKinematics6M90G.h](#).

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

- [include/KNI_InvKin/KatanaKinematics6M90G.h](#)

11.36 KNI::KatanaKinematics6M90G::position Struct Reference

Public Attributes

- double [x](#)
- double [y](#)
- double [z](#)

11.36.1 Detailed Description

Definition at line 52 of file [KatanaKinematics6M90G.h](#).

11.36.2 Member Data Documentation

11.36.2.1 double [KNI::KatanaKinematics6M90G::position::x](#)

Definition at line 53 of file [KatanaKinematics6M90G.h](#).

11.36.2.2 double [KNI::KatanaKinematics6M90G::position::y](#)

Definition at line 54 of file [KatanaKinematics6M90G.h](#).

11.36.2.3 double [KNI::KatanaKinematics6M90G::position::z](#)

Definition at line 55 of file [KatanaKinematics6M90G.h](#).

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

- [include/KNI_InvKin/KatanaKinematics6M90G.h](#)

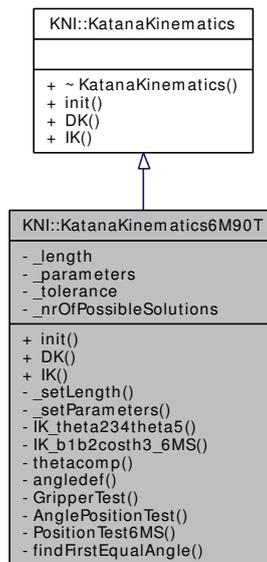
11.37 KNI::KatanaKinematics6M90T Class Reference

Author:

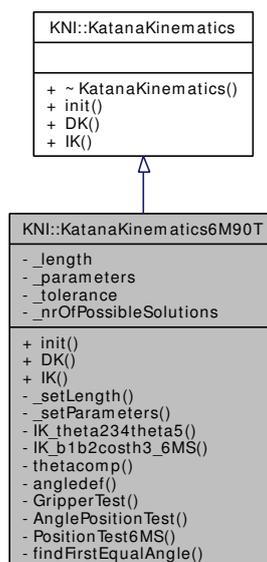
Tiziano Mueller <tiziano.mueller@neuronics.ch>

```
#include <KatanaKinematics6M90T.h>
```

Inheritance diagram for KNI::KatanaKinematics6M90T:



Collaboration diagram for KNI::KatanaKinematics6M90T:



Public Member Functions

- void `init` (`metrics` const &length, `parameter_container` const ¶meters)
- void `DK` (`coordinates` &solution, `encoders` const ¤t_encoders) const
Direct Kinematic.
- void `IK` (`encoders::iterator` solution, `coordinates` const &pose, `encoders` const &cur_angles) const
Inverse Kinematic.

Private Types

- typedef std::vector< `angles_calc` > `angles_container`

Private Member Functions

- void `_setLength` (`metrics` const &length)
- void `_setParameters` (`parameter_container` const ¶meters)
- void `IK_theta234theta5` (`angles_calc` &angle, const `position` &p_gr) const
- void `IK_b1b2costh3_6MS` (`angles_calc` &a, const `position` &p) const
- void `thetacomp` (`angles_calc` &a, const `position` &p_m, const `coordinates` &pose) const
- bool `angledef` (`angles_calc` &a) const
- bool `GripperTest` (const `position` &p_gr, const `angles_calc` &angle) const
- bool `AnglePositionTest` (const `angles_calc` &a) const
- bool `PositionTest6MS` (const double &theta1, const double &theta2, const double &theta3, const double &theta234, const `position` &p) const
- double `findFirstEqualAngle` (const `angles` &v1, const `angles` &v2) const

Private Attributes

- `metrics_length`
- `parameter_container_parameters`

Static Private Attributes

- static const double `_tolerance`
- static const int `_nrOfPossibleSolutions`

Classes

- struct `angles_calc`
- struct `position`

11.37.1 Detailed Description

Author:

Tiziano Mueller <tiziano.mueller@neuronics.ch>

Author:

Christoph Voser <christoph.voser@neuronics.ch>

Definition at line 39 of file KatanaKinematics6M90T.h.

11.37.2 Member Typedef Documentation

11.37.2.1 `typedef std::vector<angles_calc> KNI::KatanaKinematics6M90T::angles_container`
`[private]`

Definition at line 71 of file KatanaKinematics6M90T.h.

11.37.3 Member Function Documentation

11.37.3.1 `void KNI::KatanaKinematics6M90T::init (metrics const & length, parameter_container const & parameters)`

11.37.3.2 `void KNI::KatanaKinematics6M90T::DK (coordinates & solution, encoders const & current_encoders) const` `[virtual]`

Direct Kinematic.

Calculates the actual position in cartesian coordinates using the given encoders

Parameters:

solution This is where the algorithm will store the solution to (in cartesian coordinates)

current_encoders The encoder values which are being used for the calculation

Note:

strong guarantee provided

Implements [KNI::KatanaKinematics](#).

11.37.3.3 `void KNI::KatanaKinematics6M90T::IK (encoders::iterator solution, coordinates const & pose, encoders const & cur_angles) const` `[virtual]`

Inverse Kinematic.

Calculates one set of encoders (=one solution) for the given cartesian coordinates. You also have to provide the current encoders to allow the algorithm to choose between different valid solutions.

Parameters:

solution This is where the algorithm will store the solution to (in encoders)

pose The target position in cartesian coordinates plus the euler angles for the direction of the gripper

cur_angles The current angles (in encoders) of the robot

Note:

strong guarantee provided

Implements [KNI::KatanaKinematics](#).

11.37.3.4 void [KNI::KatanaKinematics6M90T::_setLength](#) ([metrics](#) const & *length*) [*inline*, *private*]

Definition at line 79 of file [KatanaKinematics6M90T.h](#).

11.37.3.5 void [KNI::KatanaKinematics6M90T::_setParameters](#) ([parameter_container](#) const & *parameters*) [*inline*, *private*]

Definition at line 80 of file [KatanaKinematics6M90T.h](#).

11.37.3.6 void [KNI::KatanaKinematics6M90T::IK_theta234theta5](#) ([angles_calc](#) & *angle*, const [position](#) & *p_gr*) const [*private*]

11.37.3.7 void [KNI::KatanaKinematics6M90T::IK_b1b2costh3_6MS](#) ([angles_calc](#) & *a*, const [position](#) & *p*) const [*private*]

11.37.3.8 void [KNI::KatanaKinematics6M90T::thetacomp](#) ([angles_calc](#) & *a*, const [position](#) & *p_m*, const [coordinates](#) & *pose*) const [*private*]

11.37.3.9 bool [KNI::KatanaKinematics6M90T::angledef](#) ([angles_calc](#) & *a*) const [*private*]

11.37.3.10 bool [KNI::KatanaKinematics6M90T::GripperTest](#) (const [position](#) & *p_gr*, const [angles_calc](#) & *angle*) const [*private*]

11.37.3.11 bool [KNI::KatanaKinematics6M90T::AnglePositionTest](#) (const [angles_calc](#) & *a*) const [*private*]

11.37.3.12 bool [KNI::KatanaKinematics6M90T::PositionTest6MS](#) (const double & *theta1*, const double & *theta2*, const double & *theta3*, const double & *theta234*, const [position](#) & *p*) const [*private*]

11.37.3.13 double [KNI::KatanaKinematics6M90T::findFirstEqualAngle](#) (const [angles](#) & *v1*, const [angles](#) & *v2*) const [*private*]

11.37.4 Member Data Documentation

11.37.4.1 [metrics](#) [KNI::KatanaKinematics6M90T::_length](#) [*private*]

Definition at line 73 of file [KatanaKinematics6M90T.h](#).

11.37.4.2 [parameter_container](#) [KNI::KatanaKinematics6M90T::_parameters](#) [*private*]

Definition at line 74 of file [KatanaKinematics6M90T.h](#).

11.37.4.3 `const double` [KNI::KatanaKinematics6M90T::_tolerance](#) `[static, private]`

Definition at line 76 of file `KatanaKinematics6M90T.h`.

11.37.4.4 `const int` [KNI::KatanaKinematics6M90T::_nrOfPossibleSolutions](#) `[static, private]`

Definition at line 77 of file `KatanaKinematics6M90T.h`.

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

- `include/KNI_InvKin/KatanaKinematics6M90T.h`

11.38 KNI::KatanaKinematics6M90T::angles_calc Struct Reference

Public Attributes

- double [theta1](#)
- double [theta2](#)
- double [theta3](#)
- double [theta4](#)
- double [theta5](#)
- double [theta6](#)
- double [theta234](#)
- double [b1](#)
- double [b2](#)
- double [costh3](#)

11.38.1 Detailed Description

Definition at line 58 of file KatanaKinematics6M90T.h.

11.38.2 Member Data Documentation

11.38.2.1 double [KNI::KatanaKinematics6M90T::angles_calc::theta1](#)

Definition at line 59 of file KatanaKinematics6M90T.h.

11.38.2.2 double [KNI::KatanaKinematics6M90T::angles_calc::theta2](#)

Definition at line 60 of file KatanaKinematics6M90T.h.

11.38.2.3 double [KNI::KatanaKinematics6M90T::angles_calc::theta3](#)

Definition at line 61 of file KatanaKinematics6M90T.h.

11.38.2.4 double [KNI::KatanaKinematics6M90T::angles_calc::theta4](#)

Definition at line 62 of file KatanaKinematics6M90T.h.

11.38.2.5 double [KNI::KatanaKinematics6M90T::angles_calc::theta5](#)

Definition at line 63 of file KatanaKinematics6M90T.h.

11.38.2.6 double [KNI::KatanaKinematics6M90T::angles_calc::theta6](#)

Definition at line 64 of file KatanaKinematics6M90T.h.

11.38.2.7 double [KNI::KatanaKinematics6M90T::angles_calc::theta234](#)

Definition at line 65 of file [KatanaKinematics6M90T.h](#).

11.38.2.8 double [KNI::KatanaKinematics6M90T::angles_calc::b1](#)

Definition at line 66 of file [KatanaKinematics6M90T.h](#).

11.38.2.9 double [KNI::KatanaKinematics6M90T::angles_calc::b2](#)

Definition at line 67 of file [KatanaKinematics6M90T.h](#).

11.38.2.10 double [KNI::KatanaKinematics6M90T::angles_calc::costh3](#)

Definition at line 68 of file [KatanaKinematics6M90T.h](#).

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

- [include/KNI_InvKin/KatanaKinematics6M90T.h](#)

11.39 KNI::KatanaKinematics6M90T::position Struct Reference

Public Attributes

- double [x](#)
- double [y](#)
- double [z](#)

11.39.1 Detailed Description

Definition at line 52 of file [KatanaKinematics6M90T.h](#).

11.39.2 Member Data Documentation

11.39.2.1 double [KNI::KatanaKinematics6M90T::position::x](#)

Definition at line 53 of file [KatanaKinematics6M90T.h](#).

11.39.2.2 double [KNI::KatanaKinematics6M90T::position::y](#)

Definition at line 54 of file [KatanaKinematics6M90T.h](#).

11.39.2.3 double [KNI::KatanaKinematics6M90T::position::z](#)

Definition at line 55 of file [KatanaKinematics6M90T.h](#).

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

- [include/KNI_InvKin/KatanaKinematics6M90T.h](#)

11.40 KNI::KinematicParameters Struct Reference

To pass different parameters for the kinematic implementations.

```
#include <KatanaKinematics.h>
```

Public Attributes

- double [angleOffset](#)
- double [angleStop](#)
- int [epc](#)
- int [encOffset](#)
- int [rotDir](#)

11.40.1 Detailed Description

To pass different parameters for the kinematic implementations.

These parameters are used for "reducing" different solutions to valid angles and to check angles against given limits (angleOffset, angleStop)

Definition at line 53 of file KatanaKinematics.h.

11.40.2 Member Data Documentation

11.40.2.1 double [KNI::KinematicParameters::angleOffset](#)

Definition at line 54 of file KatanaKinematics.h.

11.40.2.2 double [KNI::KinematicParameters::angleStop](#)

Definition at line 55 of file KatanaKinematics.h.

11.40.2.3 int [KNI::KinematicParameters::epc](#)

Definition at line 56 of file KatanaKinematics.h.

11.40.2.4 int [KNI::KinematicParameters::encOffset](#)

Definition at line 57 of file KatanaKinematics.h.

11.40.2.5 int [KNI::KinematicParameters::rotDir](#)

Definition at line 58 of file KatanaKinematics.h.

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

- include/KNI_InvKin/[KatanaKinematics.h](#)

11.41 KNI::KinematicsDefaultEncMinAlgorithm Struct Reference

```
#include <KatanaKinematicsDecisionAlgorithms.h>
```

Public Types

- typedef std::vector< int > [encoders](#)
- typedef encoders::const_iterator [c_iter](#)
- typedef std::vector< [encoders](#) >::const_iterator [t_iter](#)

Public Member Functions

- [t_iter operator\(\)](#) ([t_iter](#) targetEnc_begin, [t_iter](#) targetEnc_end, [c_iter](#) currentEnc_begin, [c_iter](#) currentEnc_end)

11.41.1 Detailed Description

Definition at line 30 of file [KatanaKinematicsDecisionAlgorithms.h](#).

11.41.2 Member Typedef Documentation

11.41.2.1 typedef std::vector<int> [KNI::KinematicsDefaultEncMinAlgorithm::encoders](#)

Definition at line 31 of file [KatanaKinematicsDecisionAlgorithms.h](#).

11.41.2.2 typedef encoders::const_iterator [KNI::KinematicsDefaultEncMinAlgorithm::c_iter](#)

Definition at line 32 of file [KatanaKinematicsDecisionAlgorithms.h](#).

11.41.2.3 typedef std::vector< [encoders](#) >::const_iterator [KNI::KinematicsDefaultEncMinAlgorithm::t_iter](#)

Definition at line 33 of file [KatanaKinematicsDecisionAlgorithms.h](#).

11.41.3 Member Function Documentation

11.41.3.1 [t_iter](#) [KNI::KinematicsDefaultEncMinAlgorithm::operator\(\)](#) ([t_iter](#) targetEnc_begin, [t_iter](#) targetEnc_end, [c_iter](#) currentEnc_begin, [c_iter](#) currentEnc_end)

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

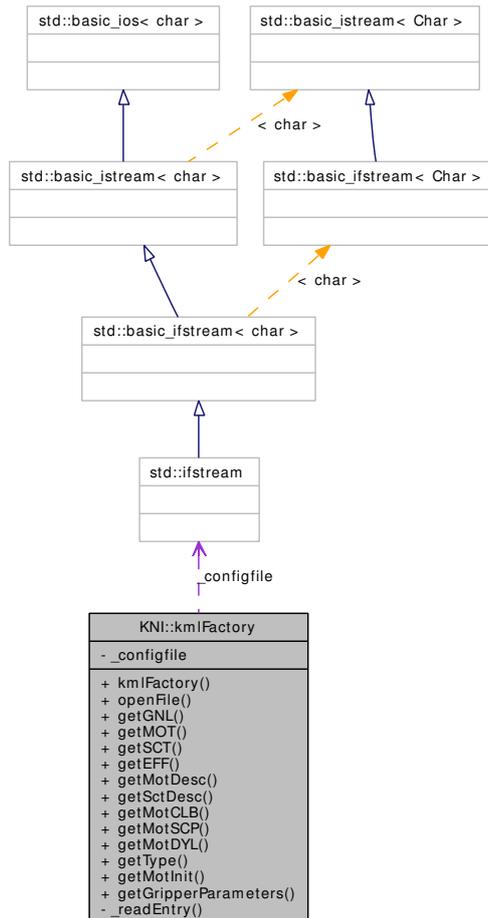
- [include/KNI_InvKin/KatanaKinematicsDecisionAlgorithms.h](#)

11.42 KNI::kmlFactory Class Reference

This class is for internal use only It may change at any time It shields the configuration file parsing.

```
#include <kmlFactories.h>
```

Collaboration diagram for KNI::kmlFactory:



Public Member Functions

- `kmlFactory ()`
- `bool openFile (const char *filepath)`
- `TKatGNL getGNL ()`
- `TKatMOT getMOT ()`
- `TKatSCT getSCT ()`
- `TKatEFF getEFF ()`
- `TMotDesc * getMotDesc (short count)`
- `TSctDesc * getSctDesc (short count)`
- `TMotCLB getMotCLB (short number)`
- `TMotSCP getMotSCP (short number)`
- `TMotDYL getMotDYL (short number)`
- `int getType ()`

returns the Katana type

- [TMotInit getMotInit](#) (short number)
- void [getGripperParameters](#) (bool &isPresent, int &openEncoders, int &closeEncoders)

Private Member Functions

- void [_readEntry](#) (char *dest, int destsz, const char *section, const char *subsection, const char *entry)

Private Attributes

- std::ifstream [_configfile](#)

11.42.1 Detailed Description

This class is for internal use only It may change at any time It shields the configuration file parsing.
Definition at line 75 of file kmlFactories.h.

11.42.2 Constructor & Destructor Documentation

11.42.2.1 KNI::kmlFactory::kmlFactory ()

11.42.3 Member Function Documentation

11.42.3.1 void KNI::kmlFactory::_readEntry (char * *dest*, int *destsz*, const char * *section*, const char * *subsection*, const char * *entry*) [private]

11.42.3.2 bool KNI::kmlFactory::openFile (const char * *filepath*) [inline]

Definition at line 83 of file kmlFactories.h.

- 11.42.3.3 [TKatGNL](#) KNI::kmlFactory::getGNL ()
- 11.42.3.4 [TKatMOT](#) KNI::kmlFactory::getMOT ()
- 11.42.3.5 [TKatSCT](#) KNI::kmlFactory::getSCT ()
- 11.42.3.6 [TKatEFF](#) KNI::kmlFactory::getEFF ()
- 11.42.3.7 [TMotDesc*](#) KNI::kmlFactory::getMotDesc (*short count*)
- 11.42.3.8 [TSctDesc*](#) KNI::kmlFactory::getSctDesc (*short count*)
- 11.42.3.9 [TMotCLB](#) KNI::kmlFactory::getMotCLB (*short number*)
- 11.42.3.10 [TMotSCP](#) KNI::kmlFactory::getMotSCP (*short number*)
- 11.42.3.11 [TMotDYL](#) KNI::kmlFactory::getMotDYL (*short number*)
- 11.42.3.12 `int` KNI::kmlFactory::getType ()

returns the Katana type

Returns:

300 for Katana300, 400 for Katana400

- 11.42.3.13 [TMotInit](#) KNI::kmlFactory::getMotInit (*short number*)
- 11.42.3.14 `void` KNI::kmlFactory::getGripperParameters (*bool & isPresent, int & openEncoders, int & closeEncoders*)

11.42.4 Member Data Documentation

- 11.42.4.1 `std::ifstream` [KNI::kmlFactory::_configfile](#) [*private*]

Definition at line 77 of file `kmlFactories.h`.

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

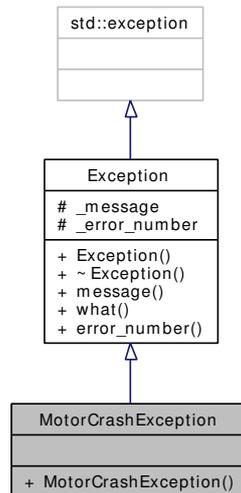
- [include/KNI/kmlFactories.h](#)

11.43 MotorCrashException Class Reference

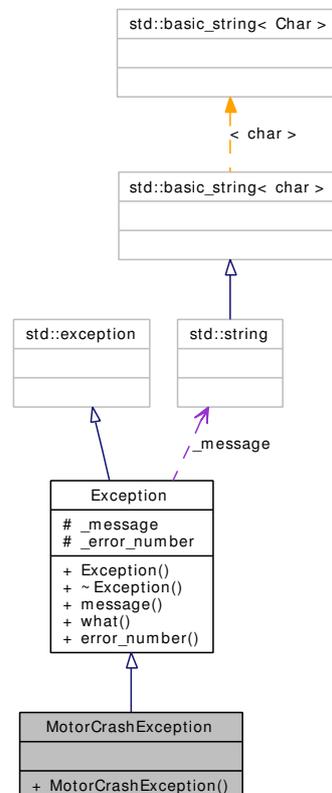
The requested motor crashed during the movement.

```
#include <kmlCommon.h>
```

Inheritance diagram for MotorCrashException:



Collaboration diagram for MotorCrashException:



Public Member Functions

- [MotorCrashException](#) () throw ()

11.43.1 Detailed Description

The requested motor crashed during the movement.

Note:

error_number=-37

Definition at line 89 of file [kmlCommon.h](#).

11.43.2 Constructor & Destructor Documentation

11.43.2.1 [MotorCrashException::MotorCrashException \(\) throw \(\)](#) [inline]

Definition at line 91 of file [kmlCommon.h](#).

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

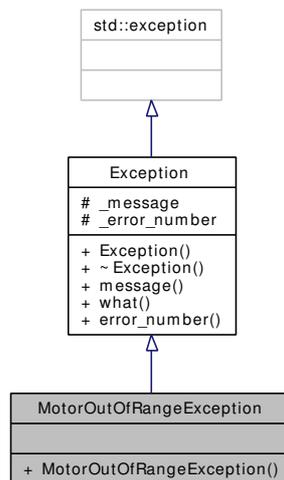
- [include/KNI/kmlCommon.h](#)

11.44 MotorOutOfRangeException Class Reference

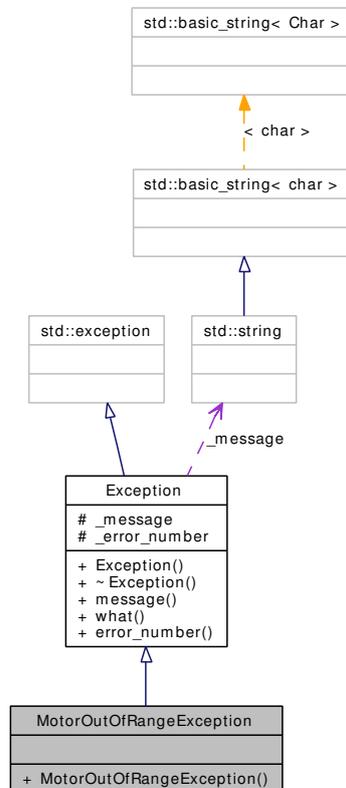
The encoders for the given motor were out of range.

```
#include <kmlCommon.h>
```

Inheritance diagram for MotorOutOfRangeException:



Collaboration diagram for MotorOutOfRangeException:



Public Member Functions

- [MotorOutOfRangeException](#) () throw ()

11.44.1 Detailed Description

The encoders for the given motor were out of range.

Note:

error_number=-35

Definition at line 71 of file `kmlCommon.h`.

11.44.2 Constructor & Destructor Documentation

11.44.2.1 `MotorOutOfRangeException::MotorOutOfRangeException () throw ()` [inline]

Definition at line 73 of file `kmlCommon.h`.

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

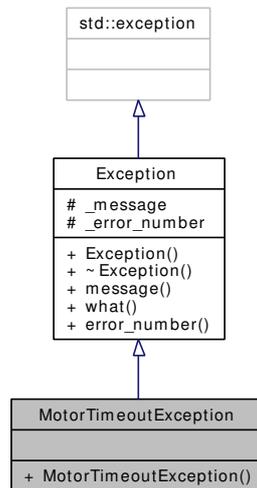
- `include/KNI/kmlCommon.h`

11.45 MotorTimeoutException Class Reference

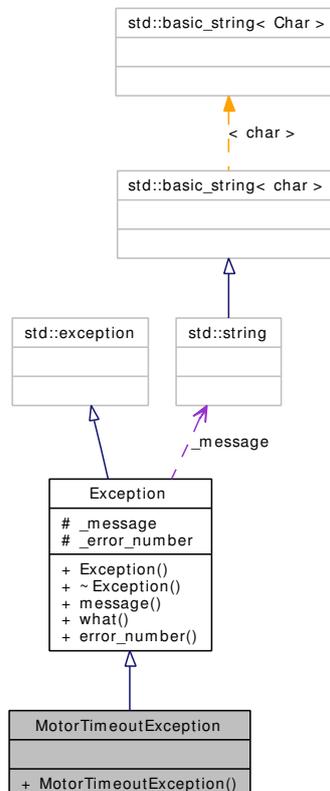
The timeout elapsed for the given motor and target position.

```
#include <kmlCommon.h>
```

Inheritance diagram for MotorTimeoutException:



Collaboration diagram for MotorTimeoutException:



Public Member Functions

- [MotorTimeoutException](#) () throw ()

11.45.1 Detailed Description

The timeout elapsed for the given motor and target position.

Note:

error_number=-36

Definition at line 80 of file [kmlCommon.h](#).

11.45.2 Constructor & Destructor Documentation

11.45.2.1 [MotorTimeoutException::MotorTimeoutException](#) () throw () [inline]

Definition at line 82 of file [kmlCommon.h](#).

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

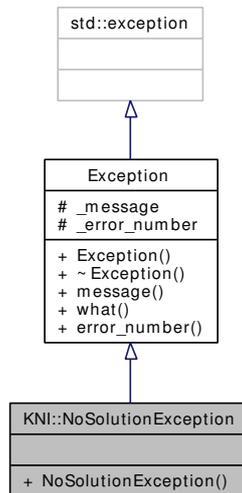
- [include/KNI/kmlCommon.h](#)

11.46 KNI::NoSolutionException Class Reference

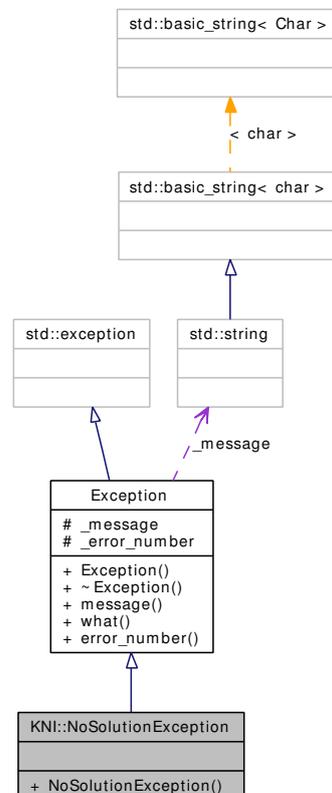
No solution found for the given cartesian coordinates.

```
#include <KatanaKinematics.h>
```

Inheritance diagram for KNI::NoSolutionException:



Collaboration diagram for KNI::NoSolutionException:



Public Member Functions

- [NoSolutionException](#) () throw ()

11.46.1 Detailed Description

No solution found for the given cartesian coordinates.

Note:

error_number=-60

Definition at line 39 of file [KatanaKinematics.h](#).

11.46.2 Constructor & Destructor Documentation

11.46.2.1 KNI::NoSolutionException::NoSolutionException () throw () [inline]

Definition at line 41 of file [KatanaKinematics.h](#).

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

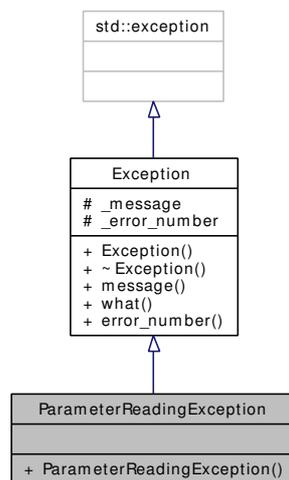
- [include/KNI_InvKin/KatanaKinematics.h](#)

11.47 ParameterReadingException Class Reference

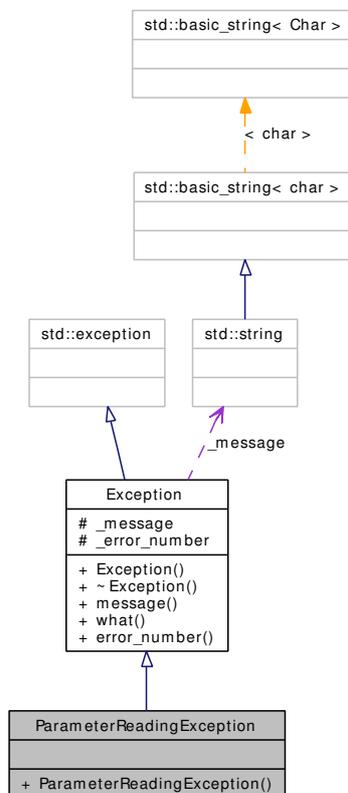
There was an error while reading a parameter from the robot.

```
#include <kmlCommon.h>
```

Inheritance diagram for ParameterReadingException:



Collaboration diagram for ParameterReadingException:



Public Member Functions

- [ParameterReadingException](#) (const std::string ¶) throw ()

11.47.1 Detailed Description

There was an error while reading a parameter from the robot.

Note:

```
error_number=-32
```

Definition at line 44 of file kmlCommon.h.

11.47.2 Constructor & Destructor Documentation

11.47.2.1 ParameterReadingException::ParameterReadingException (const std::string & para) throw () [inline]

Definition at line 46 of file kmlCommon.h.

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

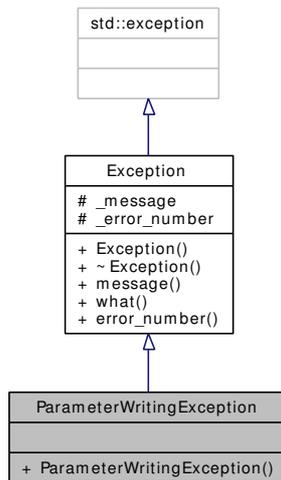
- [include/KNI/kmlCommon.h](#)

11.48 ParameterWritingException Class Reference

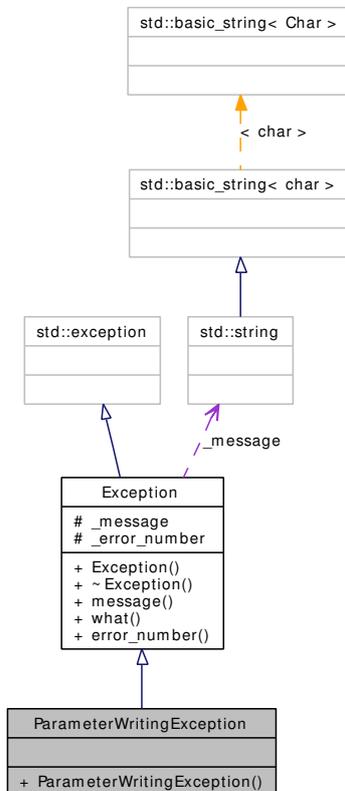
The data you wanted to send to the robot was invalid.

```
#include <kmlCommon.h>
```

Inheritance diagram for ParameterWritingException:



Collaboration diagram for ParameterWritingException:



Public Member Functions

- [ParameterWritingException](#) (const std::string ¶) throw ()

11.48.1 Detailed Description

The data you wanted to send to the robot was invalid.

Note:

```
error_number=-33
```

Definition at line 53 of file kmlCommon.h.

11.48.2 Constructor & Destructor Documentation

11.48.2.1 ParameterWritingException::ParameterWritingException (const std::string & para) throw () [inline]

Definition at line 55 of file kmlCommon.h.

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

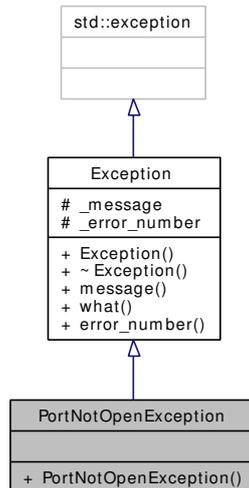
- [include/KNI/kmlCommon.h](#)

11.49 PortNotOpenException Class Reference

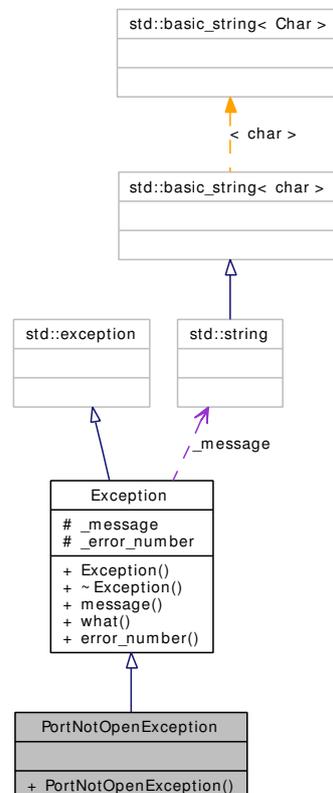
The port was not open.

```
#include <cdlCOMExceptions.h>
```

Inheritance diagram for PortNotOpenException:



Collaboration diagram for PortNotOpenException:



Public Member Functions

- [PortNotOpenException](#) (const std::string &port) throw ()

11.49.1 Detailed Description

The port was not open.

Note:

error_number=-12

Definition at line 65 of file `cdlCOMExceptions.h`.

11.49.2 Constructor & Destructor Documentation

11.49.2.1 `PortNotOpenException::PortNotOpenException` (const std::string & *port*) throw () [inline]

Definition at line 67 of file `cdlCOMExceptions.h`.

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

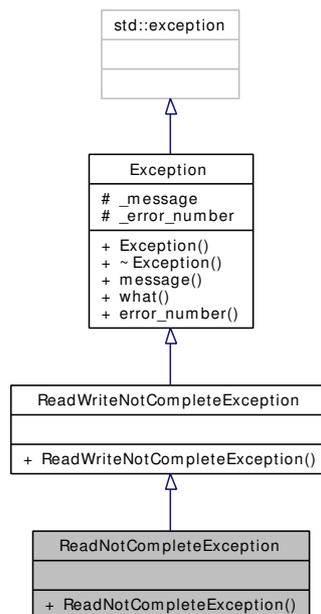
- `include/KNI/cdlCOMExceptions.h`

11.50 ReadNotCompleteException Class Reference

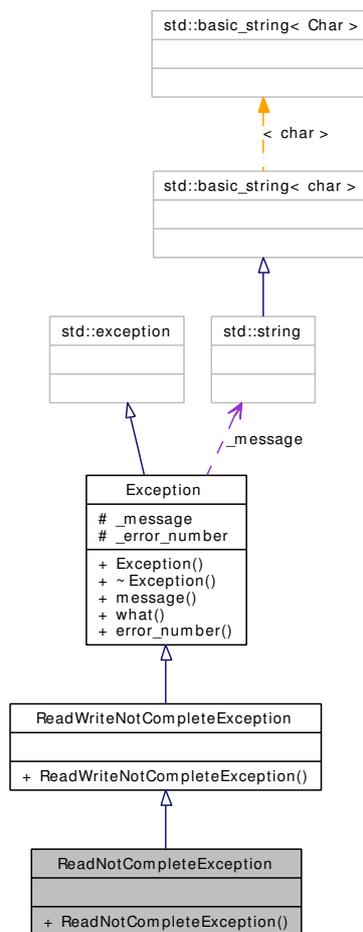
The Katana didn't answer correctly within the given timeout.

```
#include <cdlCOMExceptions.h>
```

Inheritance diagram for ReadNotCompleteException:



Collaboration diagram for ReadNotCompleteException:



Public Member Functions

- [ReadNotCompleteException](#) (const std::string &port) throw ()

11.50.1 Detailed Description

The Katana didn't answer correctly within the given timeout.

Note:

error_number=-16

Definition at line 112 of file cdICOMExceptions.h.

11.50.2 Constructor & Destructor Documentation

11.50.2.1 ReadNotCompleteException::ReadNotCompleteException (const std::string & port) throw () [inline]

Definition at line 114 of file cdICOMExceptions.h.

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

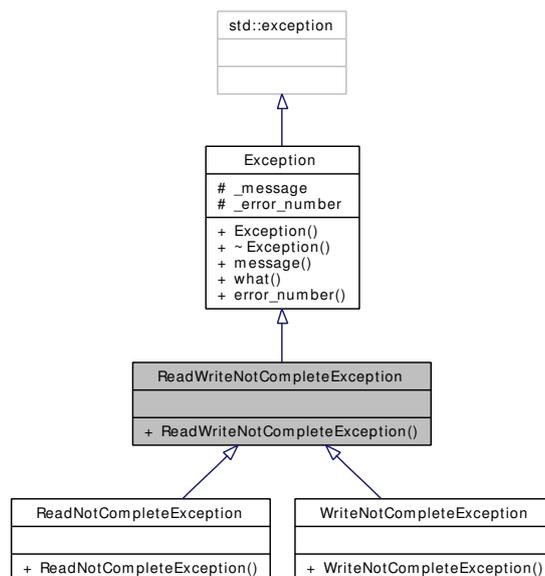
- [include/KNI/cdlCOMExceptions.h](#)

11.51 ReadWriteNotCompleteException Class Reference

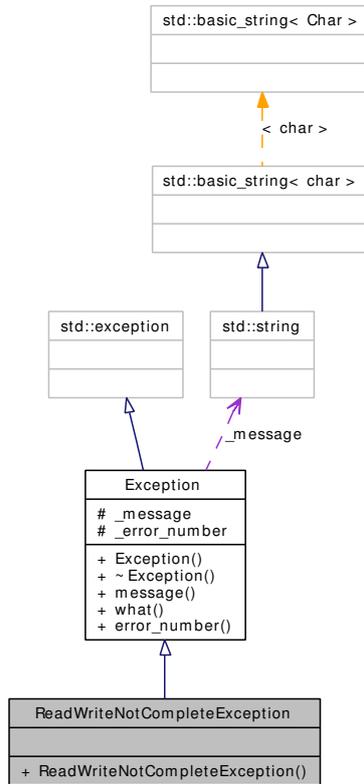
This exception is the base for the WriteNotComplete and [ReadNotCompleteException](#).

```
#include <cdlCOMExceptions.h>
```

Inheritance diagram for ReadWriteNotCompleteException:



Collaboration diagram for ReadWriteNotCompleteException:



Public Member Functions

- [ReadWriteNotCompleteException](#) (const std::string &errstr, const int error_number) throw ()

11.51.1 Detailed Description

This exception is the base for the WriteNotComplete and [ReadNotCompleteException](#).

Definition at line 94 of file `cdlCOMExceptions.h`.

11.51.2 Constructor & Destructor Documentation

11.51.2.1 [ReadWriteNotCompleteException::ReadWriteNotCompleteException](#) (const std::string & *errstr*, const int *error_number*) throw () [inline]

Definition at line 96 of file `cdlCOMExceptions.h`.

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

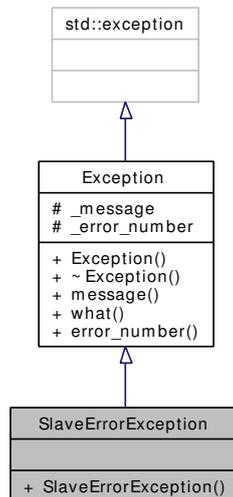
- [include/KNI/cdlCOMExceptions.h](#)

11.52 SlaveErrorException Class Reference

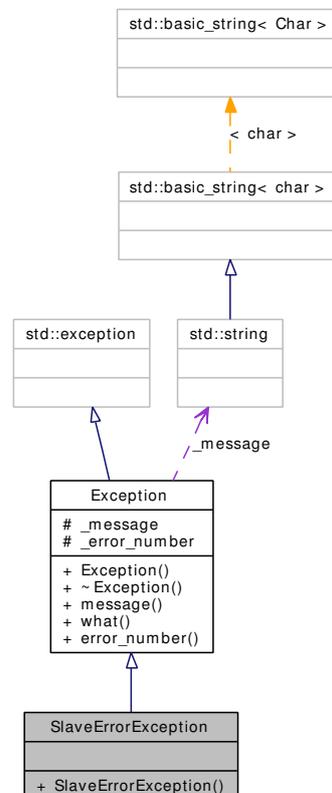
Slave error occurred.

```
#include <kmlCommon.h>
```

Inheritance diagram for SlaveErrorException:



Collaboration diagram for SlaveErrorException:



Public Member Functions

- [SlaveErrorException](#) () throw ()

11.52.1 Detailed Description

Slave error occurred.

Note:

error_number=-31

Definition at line 35 of file kmlCommon.h.

11.52.2 Constructor & Destructor Documentation

11.52.2.1 [SlaveErrorException::SlaveErrorException \(\) throw \(\)](#) [inline]

Definition at line 37 of file kmlCommon.h.

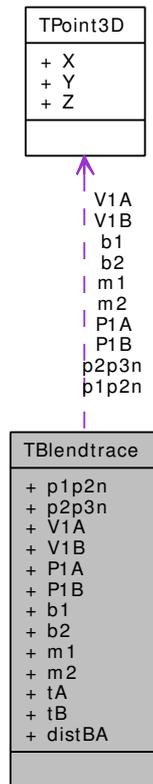
The documentation for this class was generated from the following file:

- include/KNI/[kmlCommon.h](#)

11.53 TBlendtrace Struct Reference

```
#include <lmBase.h>
```

Collaboration diagram for TBlendtrace:



Public Attributes

- [TPoint3D p1p2n](#)
- [TPoint3D p2p3n](#)
- [TPoint3D V1A](#)
- [TPoint3D V1B](#)
- [TPoint3D P1A](#)
- [TPoint3D P1B](#)
- [TPoint3D b1](#)
- [TPoint3D b2](#)
- [TPoint3D m1](#)
- [TPoint3D m2](#)
- [double tA](#)
- [double tB](#)
- [double distBA](#)

11.53.1 Detailed Description

Definition at line 83 of file lmBase.h.

11.53.2 Member Data Documentation

11.53.2.1 [TPoint3D TBlendtrace::p1p2n](#)

Definition at line 84 of file lmBase.h.

11.53.2.2 [TPoint3D TBlendtrace::p2p3n](#)

Definition at line 85 of file lmBase.h.

11.53.2.3 [TPoint3D TBlendtrace::V1A](#)

Definition at line 86 of file lmBase.h.

11.53.2.4 [TPoint3D TBlendtrace::V1B](#)

Definition at line 87 of file lmBase.h.

11.53.2.5 [TPoint3D TBlendtrace::P1A](#)

Definition at line 88 of file lmBase.h.

11.53.2.6 [TPoint3D TBlendtrace::P1B](#)

Definition at line 89 of file lmBase.h.

11.53.2.7 [TPoint3D TBlendtrace::b1](#)

Definition at line 90 of file lmBase.h.

11.53.2.8 [TPoint3D TBlendtrace::b2](#)

Definition at line 91 of file lmBase.h.

11.53.2.9 [TPoint3D TBlendtrace::m1](#)

Definition at line 92 of file lmBase.h.

11.53.2.10 [TPoint3D TBlendtrace::m2](#)

Definition at line 93 of file lmBase.h.

11.53.2.11 [double TBlendtrace::tA](#)

Definition at line 94 of file lmBase.h.

11.53.2.12 double TBlendtrace::tB

Definition at line 95 of file lmBase.h.

11.53.2.13 double TBlendtrace::distBA

Definition at line 96 of file lmBase.h.

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

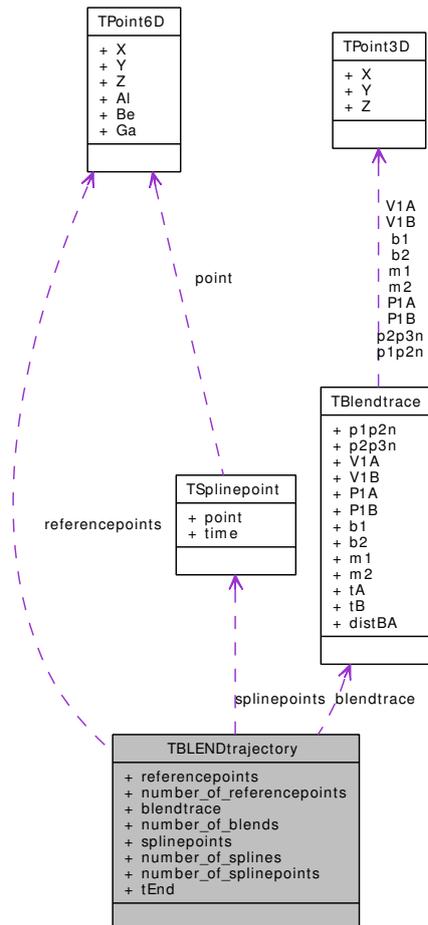
- [include/KNI_LM/lmBase.h](#)

11.54 TBLENDtrajectory Struct Reference

[LMBLEND] Trajectory points

```
#include <lmBase.h>
```

Collaboration diagram for TBLENDtrajectory:



Public Attributes

- [TPoint6D](#) * `referencepoints`
- short `number_of_referencepoints`
- [TBlendtrace](#) * `blendtrace`
- short `number_of_blends`
- [TSplinepoint](#) * `splinepoints`
- short `number_of_splines`
- short `number_of_splinepoints`
- double `tEnd`

11.54.1 Detailed Description

[LMBLEND] Trajectory points

Definition at line 107 of file lmBase.h.

11.54.2 Member Data Documentation

11.54.2.1 [TPoint6D*](#) [TBLENDtrajectory::referencepoints](#)

Definition at line 108 of file lmBase.h.

11.54.2.2 [short](#) [TBLENDtrajectory::number_of_referencepoints](#)

Definition at line 109 of file lmBase.h.

11.54.2.3 [TBlendtrace*](#) [TBLENDtrajectory::blendtrace](#)

Definition at line 110 of file lmBase.h.

11.54.2.4 [short](#) [TBLENDtrajectory::number_of_blends](#)

Definition at line 111 of file lmBase.h.

11.54.2.5 [TSplinepoint*](#) [TBLENDtrajectory::splinepoints](#)

Definition at line 112 of file lmBase.h.

11.54.2.6 [short](#) [TBLENDtrajectory::number_of_splines](#)

Definition at line 113 of file lmBase.h.

11.54.2.7 [short](#) [TBLENDtrajectory::number_of_splinepoints](#)

Definition at line 114 of file lmBase.h.

11.54.2.8 [double](#) [TBLENDtrajectory::tEnd](#)

Definition at line 115 of file lmBase.h.

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

- [include/KNI_LM/lmBase.h](#)

11.55 TCdlCOMDesc Struct Reference

This structrue stores the attributes for a serial port device.

```
#include <cdlCOM.h>
```

Public Attributes

- int [port](#)
serial port number
- int [baud](#)
baud rate of port
- int [data](#)
data bit
- int [parity](#)
parity bit
- int [stop](#)
stop bit
- int [rttc](#)
read total timeout
- int [wttc](#)
write total timeout

11.55.1 Detailed Description

This structrue stores the attributes for a serial port device.

Definition at line 53 of file cdlCOM.h.

11.55.2 Member Data Documentation

11.55.2.1 int [TCdlCOMDesc::port](#)

serial port number

Definition at line 54 of file cdlCOM.h.

11.55.2.2 int [TCdlCOMDesc::baud](#)

baud rate of port

Definition at line 55 of file cdlCOM.h.

11.55.2.3 int TCdlCOMDesc::data

data bit

Definition at line 56 of file cdCOM.h.

11.55.2.4 int TCdlCOMDesc::parity

parity bit

Definition at line 57 of file cdCOM.h.

11.55.2.5 int TCdlCOMDesc::stop

stop bit

Definition at line 58 of file cdCOM.h.

11.55.2.6 int TCdlCOMDesc::rttc

read total timeout

Definition at line 59 of file cdCOM.h.

11.55.2.7 int TCdlCOMDesc::wttc

write total timeout

Definition at line 60 of file cdCOM.h.

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

- [include/KNI/cdCOM.h](#)

11.56 THeader Struct Reference

Header of a communication packet.

```
#include <cplSerial.h>
```

Public Attributes

- [byte size](#)
header size
- [byte data \[256\]](#)
data part: 16x zero, 1x one, 1x katadr

11.56.1 Detailed Description

Header of a communication packet.

Definition at line 56 of file cplSerial.h.

11.56.2 Member Data Documentation

11.56.2.1 [byte THeader::size](#)

header size

Definition at line 57 of file cplSerial.h.

11.56.2.2 [byte THeader::data\[256\]](#)

data part: 16x zero, 1x one, 1x katadr

Definition at line 58 of file cplSerial.h.

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

- [include/KNI/cplSerial.h](#)

11.57 KNI::Timer Class Reference

Provides a stop-watch-like class with a resolution of milliseconds.

```
#include <Timer.h>
```

Public Member Functions

- [Timer](#) ()
- [Timer](#) (long timeout)
- void [Set](#) (long timeout)
- void [Start](#) ()
- void [Set_And_Start](#) (long timeout)
- bool [Elapsed](#) () const

Returns true if timer is elapsed.

- long [ElapsedTime](#) () const

Returns the elapsed time.

- void [WaitUntilElapsed](#) () const

Block until time's up.

Private Member Functions

- long [_ElapsedTime](#) () const

Platform specific implementation of [ElapsedTime\(\)](#).

Private Attributes

- long [_timeout](#)
- timeval [_ct](#)

11.57.1 Detailed Description

Provides a stop-watch-like class with a resolution of milliseconds.

Definition at line 41 of file `Timer.h`.

11.57.2 Constructor & Destructor Documentation

11.57.2.1 `KNI::Timer::Timer ()`

11.57.2.2 `KNI::Timer::Timer (long timeout)`

11.57.3 Member Function Documentation

11.57.3.1 `long KNI::Timer::_ElapsedTime () const` [private]

Platform specific implementation of [ElapsedTime\(\)](#).

11.57.3.2 `void KNI::Timer::Set (long timeout)`

11.57.3.3 `void KNI::Timer::Start ()`

11.57.3.4 `void KNI::Timer::Set_And_Start (long timeout)`

11.57.3.5 `bool KNI::Timer::Elapsed () const`

Returns true if timer is elapsed.

11.57.3.6 `long KNI::Timer::ElapsedTime () const`

Returns the elapsed time.

11.57.3.7 `void KNI::Timer::WaitUntilElapsed () const`

Block until time's up.

11.57.4 Member Data Documentation

11.57.4.1 `long KNI::Timer::_timeout` [private]

Definition at line 43 of file [Timer.h](#).

11.57.4.2 `struct timeval KNI::Timer::_ct` [private]

Definition at line 48 of file [Timer.h](#).

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

- [include/common/Timer.h](#)

11.58 TKatCBX Struct Reference

[CBX] connector box

```
#include <kmlBase.h>
```

Public Attributes

- bool [inp](#) [2]
input: green & red LED
- bool [out](#) [2]
output: green & red LED

11.58.1 Detailed Description

[CBX] connector box

Definition at line 93 of file kmlBase.h.

11.58.2 Member Data Documentation

11.58.2.1 bool [TKatCBX::inp](#)[2]

input: green & red LED

Definition at line 94 of file kmlBase.h.

11.58.2.2 bool [TKatCBX::out](#)[2]

output: green & red LED

Definition at line 95 of file kmlBase.h.

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

- [include/KNI/kmlBase.h](#)

11.59 TKatCTB Struct Reference

[CTB] command table defined in the firmware

```
#include <kmlBase.h>
```

Public Attributes

- [byte cmdtbl](#) [256]
command table

11.59.1 Detailed Description

[CTB] command table defined in the firmware

Definition at line 87 of file kmlBase.h.

11.59.2 Member Data Documentation

11.59.2.1 [byte TKatCTB::cmdtbl](#)[256]

command table

Definition at line 88 of file kmlBase.h.

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

- [include/KNI/kmlBase.h](#)

11.60 TKatECH Struct Reference

[ECH] echo

```
#include <kmlBase.h>
```

Public Attributes

- [byte echo](#)
echo answer

11.60.1 Detailed Description

[ECH] echo

Definition at line 100 of file kmlBase.h.

11.60.2 Member Data Documentation

11.60.2.1 [byte TKatECH::echo](#)

echo answer

Definition at line 101 of file kmlBase.h.

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

- [include/KNI/kmlBase.h](#)

11.61 TKatEFF Struct Reference

Inverse Kinematics structure of the endeffektor.

```
#include <kmlBase.h>
```

Public Attributes

- double [arr_segment](#) [4]
length of the Katana segments

11.61.1 Detailed Description

Inverse Kinematics structure of the endeffektor.

This structure describes the properties of the endeffektor and it's used for the inverse kinematic calculations. An endeffektor is a point where the attributes of this structure belong to. Please remember that the actual inverse kinematic calculations have been set up **only** for the Katana **6M** robot! So do not be astonished if you get strange behaviour with a Katana **5M**.

Definition at line 113 of file kmlBase.h.

11.61.2 Member Data Documentation

11.61.2.1 double [TKatEFF::arr_segment](#)[4]

length of the Katana segments

Definition at line 114 of file kmlBase.h.

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

- [include/KNI/kmlBase.h](#)

11.62 TKatGNL Struct Reference

[GNL] general robot attributes

```
#include <kmlBase.h>
```

Public Attributes

- [byte adr](#)
jumper adress
- [char modelName \[255\]](#)
model name

11.62.1 Detailed Description

[GNL] general robot attributes

Definition at line 67 of file kmlBase.h.

11.62.2 Member Data Documentation

11.62.2.1 [byte TKatGNL::adr](#)

jumper adress

Definition at line 68 of file kmlBase.h.

11.62.2.2 [char TKatGNL::modelName\[255\]](#)

model name

Definition at line 69 of file kmlBase.h.

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

- [include/KNI/kmlBase.h](#)

11.63 TKatIDS Struct Reference

[IDS] identification string

```
#include <kmlBase.h>
```

Public Attributes

- [byte strID](#) [256]
id string

11.63.1 Detailed Description

[IDS] identification string

Definition at line 81 of file kmlBase.h.

11.63.2 Member Data Documentation

11.63.2.1 [byte TKatIDS::strID](#)[256]

id string

Definition at line 82 of file kmlBase.h.

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

- [include/KNI/kmlBase.h](#)

11.64 TKatMFW Struct Reference

[MFW] master firmware version/revision number

```
#include <kmlBase.h>
```

Public Attributes

- [byte ver](#)
version
- [byte rev](#)
revision

11.64.1 Detailed Description

[MFW] master firmware version/revision number

Definition at line 74 of file kmlBase.h.

11.64.2 Member Data Documentation

11.64.2.1 [byte TKatMFW::ver](#)

version

Definition at line 75 of file kmlBase.h.

11.64.2.2 [byte TKatMFW::rev](#)

revision

Definition at line 76 of file kmlBase.h.

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

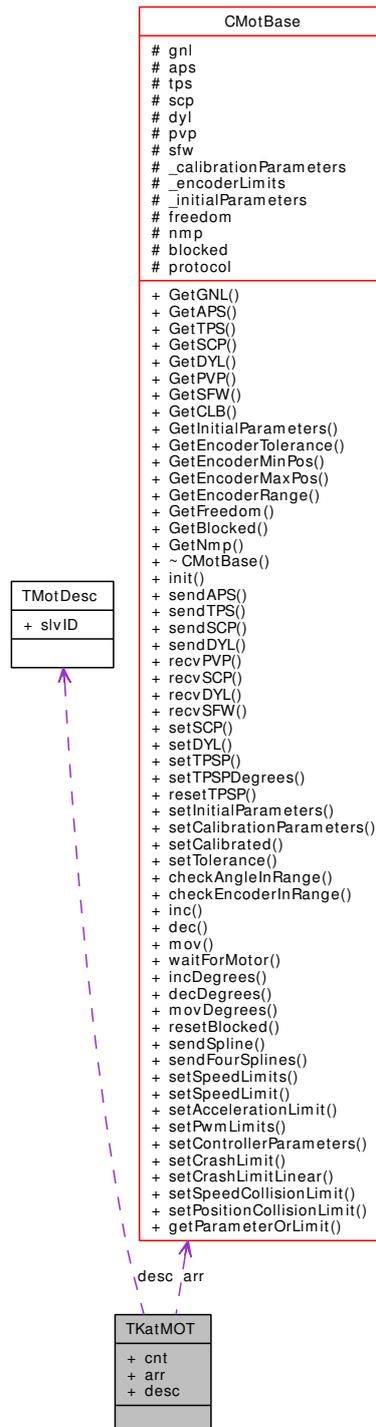
- [include/KNI/kmlBase.h](#)

11.65 TKatMOT Struct Reference

[MOT] every motor's attributes

```
#include <kmlMotBase.h>
```

Collaboration diagram for TKatMOT:



Public Attributes

- short [cnt](#)
count of motors
- [CMotBase](#) * [arr](#)
array of motors
- [TMotDesc](#) * [desc](#)
description[]

11.65.1 Detailed Description

[MOT] every motor's attributes

Definition at line 40 of file [kmlMotBase.h](#).

11.65.2 Member Data Documentation

11.65.2.1 short [TKatMOT::cnt](#)

count of motors

Definition at line 41 of file [kmlMotBase.h](#).

11.65.2.2 [CMotBase](#)* [TKatMOT::arr](#)

array of motors

Definition at line 42 of file [kmlMotBase.h](#).

11.65.2.3 [TMotDesc](#)* [TKatMOT::desc](#)

[description](#)[]

Definition at line 43 of file [kmlMotBase.h](#).

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

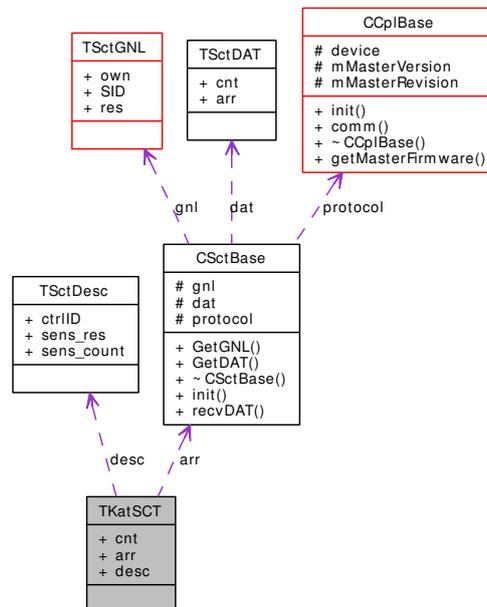
- [include/KNI/kmlMotBase.h](#)

11.66 TKatSCT Struct Reference

[SCT] every sens ctrl's attributes

```
#include <kmlSctBase.h>
```

Collaboration diagram for TKatSCT:



Public Attributes

- short `cnt`
count of sens ctrl's
- `CSctBase * arr`
array of sens ctrl's
- `TScDesc * desc`
description[]

11.66.1 Detailed Description

[SCT] every sens ctrl's attributes

Definition at line 41 of file kmlSctBase.h.

11.66.2 Member Data Documentation

11.66.2.1 short `TKatSCT::cnt`

count of sens ctrl's

Definition at line 42 of file kmlSctBase.h.

11.66.2.2 CSctBase* TKatSCT::arr

array of sens ctrl's

Definition at line 43 of file kmlSctBase.h.

11.66.2.3 TSctDesc* TKatSCT::desc

description[]

Definition at line 44 of file kmlSctBase.h.

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

- [include/KNI/kmlSctBase.h](#)

11.67 TLM_points Struct Reference

[LM] linear movement: points to be interpolated

```
#include <lmBase.h>
```

Public Attributes

- double [pos](#)
position of one point to be interpolated (% refer to the total trajectory)
- double [time](#)
time that it takes to reach the point (from starting position)

11.67.1 Detailed Description

[LM] linear movement: points to be interpolated

Definition at line 36 of file lmBase.h.

11.67.2 Member Data Documentation

11.67.2.1 double [TLM_points::pos](#)

position of one point to be interpolated (% refer to the total trajectory)

Definition at line 37 of file lmBase.h.

11.67.2.2 double [TLM_points::time](#)

time that it takes to reach the point (from starting position)

Definition at line 38 of file lmBase.h.

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

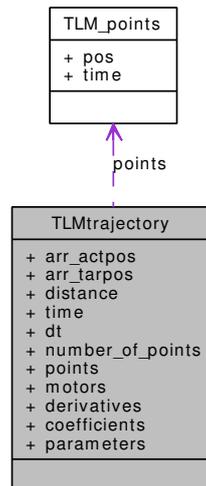
- [include/KNI_LM/lmBase.h](#)

11.68 TLMtrajectory Struct Reference

[LM] linear movement: parameters

```
#include <lmBase.h>
```

Collaboration diagram for TLMtrajectory:



Public Attributes

- double * [arr_actpos](#)
current position in cartesian units
- double * [arr_tarpos](#)
target position in cartesian units
- int [distance](#)
distance between target and current position
- double [time](#)
time that it takes from current position to target position
- double [dt](#)
time elapsed between one step and the next one
- short [number_of_points](#)
number of points to interpolate
- [TLM_points](#) * [points](#)
points to be interpolated
- short ** [motors](#)
motor position in each point to be interpolated

- double ** [derivatives](#)
second order derivatives of the polinomes that join the points, in the points
- double *** [coefficients](#)
coefficients of the polinomes that join the points
- short *** [parameters](#)
parameters to be sent in the command 'L' packet

11.68.1 Detailed Description

[LM] linear movement: parameters

Definition at line 43 of file lmBase.h.

11.68.2 Member Data Documentation

11.68.2.1 double* [TLMtrajectory::arr_actpos](#)

current position in cartesian units

Definition at line 44 of file lmBase.h.

11.68.2.2 double* [TLMtrajectory::arr_tarpos](#)

target position in cartesian units

Definition at line 45 of file lmBase.h.

11.68.2.3 int [TLMtrajectory::distance](#)

distance between target and current position

Definition at line 46 of file lmBase.h.

11.68.2.4 double [TLMtrajectory::time](#)

time that it takes from current position to target position

Definition at line 47 of file lmBase.h.

11.68.2.5 double [TLMtrajectory::dt](#)

time elapsed between one step and the next one

Definition at line 48 of file lmBase.h.

11.68.2.6 short [TLMtrajectory::number_of_points](#)

number of points to interpolate

Definition at line 49 of file lmBase.h.

11.68.2.7 [TLM_points*](#) [TLMtrajectory::points](#)

points to be interpolated

Definition at line 50 of file lmBase.h.

11.68.2.8 short** [TLMtrajectory::motors](#)

motor position in each point to be interpolated

Definition at line 51 of file lmBase.h.

11.68.2.9 double** [TLMtrajectory::derivatives](#)

second order derivatives of the polinomes that join the points, in the points

Definition at line 52 of file lmBase.h.

11.68.2.10 double*** [TLMtrajectory::coefficients](#)

coefficients of the polinomes that join the points

Definition at line 53 of file lmBase.h.

11.68.2.11 short*** [TLMtrajectory::parameters](#)

parameters to be sent in the command 'L' packet

Definition at line 54 of file lmBase.h.

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

- [include/KNI_LM/lmBase.h](#)

11.69 TMLMIP Struct Reference

[LM] Store intermediate targets for multiple linear movements

```
#include <lmBase.h>
```

Public Attributes

- short [mlm_intermediate_pos](#) [5]
current position in cartesian units

11.69.1 Detailed Description

[LM] Store intermediate targets for multiple linear movements

Definition at line 59 of file lmBase.h.

11.69.2 Member Data Documentation

11.69.2.1 short [TMLMIP::mlm_intermediate_pos](#)[5]

current position in cartesian units

Definition at line 60 of file lmBase.h.

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

- [include/KNI_LM/lmBase.h](#)

11.70 TMotAPS Struct Reference

[APS] actual position

```
#include <kmlMotBase.h>
```

Public Attributes

- [TMotCmdFlg mcfAPS](#)
motor command flag
- short [actpos](#)
actual position

11.70.1 Detailed Description

[APS] actual position

Definition at line 95 of file kmlMotBase.h.

11.70.2 Member Data Documentation

11.70.2.1 [TMotCmdFlg TMotAPS::mcfAPS](#)

motor command flag

Definition at line 96 of file kmlMotBase.h.

11.70.2.2 short [TMotAPS::actpos](#)

actual position

Definition at line 97 of file kmlMotBase.h.

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

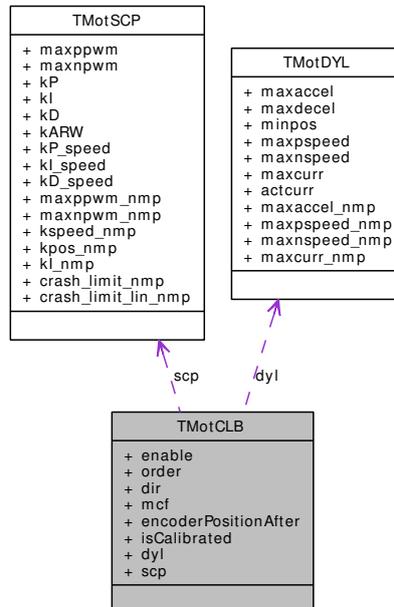
- [include/KNI/kmlMotBase.h](#)

11.71 TMotCLB Struct Reference

Calibration structure for single motors.

```
#include <kmlMotBase.h>
```

Collaboration diagram for TMotCLB:



Public Attributes

- bool [enable](#)
enable/disable
- short [order](#)
order in which this motor will be calibrated. range: 0..5
- [TSearchDir](#) [dir](#)
search direction for mech. stopper
- [TMotCmdFlg](#) [mcf](#)
motor flag after calibration
- int [encoderPositionAfter](#)
- bool [isCalibrated](#)
- [TMotDYL](#) [dyl](#)
- [TMotSCP](#) [scp](#)

11.71.1 Detailed Description

Calibration structure for single motors.

Definition at line 181 of file kmlMotBase.h.

11.71.2 Member Data Documentation

11.71.2.1 `bool TMotCLB::enable`

enable/disable

Definition at line 182 of file `kmlMotBase.h`.

11.71.2.2 `short TMotCLB::order`

order in which this motor will be calibrated. range: 0..5

Definition at line 183 of file `kmlMotBase.h`.

11.71.2.3 `TSearchDir TMotCLB::dir`

search direction for mech. stopper

Definition at line 185 of file `kmlMotBase.h`.

11.71.2.4 `TMotCmdFlg TMotCLB::mcf`

motor flag after calibration

Definition at line 186 of file `kmlMotBase.h`.

11.71.2.5 `int TMotCLB::encoderPositionAfter`

Definition at line 188 of file `kmlMotBase.h`.

11.71.2.6 `bool TMotCLB::isCalibrated`

Definition at line 189 of file `kmlMotBase.h`.

11.71.2.7 `TMotDYL TMotCLB::dyl`

Definition at line 191 of file `kmlMotBase.h`.

11.71.2.8 `TMotSCP TMotCLB::scp`

Definition at line 192 of file `kmlMotBase.h`.

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

- `include/KNI/kmlMotBase.h`

11.72 TMotDesc Struct Reference

motor description (partly)

```
#include <kmlMotBase.h>
```

Public Attributes

- [byte slvID](#)

slave number

11.72.1 Detailed Description

motor description (partly)

Definition at line 34 of file kmlMotBase.h.

11.72.2 Member Data Documentation

11.72.2.1 [byte TMotDesc::slvID](#)

slave number

Definition at line 35 of file kmlMotBase.h.

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

- [include/KNI/kmlMotBase.h](#)

11.73 TMotDYL Struct Reference

[DYL] dynamic limits

```
#include <kmlMotBase.h>
```

Public Attributes

- [byte maxaccel](#)
max acceleration
- [byte maxdecel](#)
max deceleration
- [short minpos](#)
not yet active
- [short maxpspeed](#)
max. allowed forward speed
- [short maxnspeed](#)
max. allowed reverse speed; pos!
- [byte maxcurr](#)
max current
- [byte actcurr](#)
actual current
- [byte maxaccel_nmp](#)
Maximal acceleration and deceleration.
- [short maxpspeed_nmp](#)
Max. allowed forward speed.
- [short maxnspeed_nmp](#)
Max. allowed reverse speed.
- [byte maxcurr_nmp](#)
set the maximal current

11.73.1 Detailed Description

[DYL] dynamic limits

Definition at line 137 of file kmlMotBase.h.

11.73.2 Member Data Documentation

11.73.2.1 [byte TMotDYL::maxaccel](#)

max acceleration

Definition at line 141 of file kmlMotBase.h.

11.73.2.2 [byte TMotDYL::maxdecel](#)

max deceleration

Definition at line 142 of file kmlMotBase.h.

11.73.2.3 [short TMotDYL::minpos](#)

not yet active

Definition at line 143 of file kmlMotBase.h.

11.73.2.4 [short TMotDYL::maxpspeed](#)

max. allowed forward speed

Definition at line 144 of file kmlMotBase.h.

11.73.2.5 [short TMotDYL::maxnspeed](#)

max. allowed reverse speed; pos!

Definition at line 145 of file kmlMotBase.h.

11.73.2.6 [byte TMotDYL::maxcurr](#)

max current

Definition at line 148 of file kmlMotBase.h.

11.73.2.7 [byte TMotDYL::actcurr](#)

actual current

Definition at line 149 of file kmlMotBase.h.

11.73.2.8 [byte TMotDYL::maxaccel_nmp](#)

Maximal acceleration and deceleration.

Definition at line 153 of file kmlMotBase.h.

11.73.2.9 short [TMotDYL::maxpspeed_nmp](#)

Max. allowed forward speed.

Definition at line 154 of file kmlMotBase.h.

11.73.2.10 short [TMotDYL::maxnspeed_nmp](#)

Max. allowed reverse speed.

Definition at line 155 of file kmlMotBase.h.

11.73.2.11 byte [TMotDYL::maxcurr_nmp](#)

set the maximal current

Definition at line 156 of file kmlMotBase.h.

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

- [include/KNI/kmlMotBase.h](#)

11.74 TMotENL Struct Reference

[ENL] limits in encoder values (INTERNAL STRUCTURE!)

```
#include <kmlMotBase.h>
```

Public Attributes

- int [enc_range](#)
motor's range in encoder values
- int [enc_minpos](#)
motor's minimum position in encoder values
- int [enc_maxpos](#)
motor's maximum position in encoder values
- int [enc_per_cycle](#)
number of encoder units needed to complete 360 degrees;
- int [enc_tolerance](#)
encoder units of tolerance to accept that a position has been reached

11.74.1 Detailed Description

[ENL] limits in encoder values (INTERNAL STRUCTURE!)

Definition at line 170 of file kmlMotBase.h.

11.74.2 Member Data Documentation

11.74.2.1 int [TMotENL::enc_range](#)

motor's range in encoder values

Definition at line 171 of file kmlMotBase.h.

11.74.2.2 int [TMotENL::enc_minpos](#)

motor's minimum position in encoder values

Definition at line 172 of file kmlMotBase.h.

11.74.2.3 int [TMotENL::enc_maxpos](#)

motor's maximum position in encoder values

Definition at line 173 of file kmlMotBase.h.

11.74.2.4 int [TMotENL::enc_per_cycle](#)

number of encoder units needed to complete 360 degrees;

Definition at line 174 of file [kmlMotBase.h](#).

11.74.2.5 int [TMotENL::enc_tolerance](#)

encoder units of tolerance to accept that a position has been reached

Definition at line 175 of file [kmlMotBase.h](#).

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

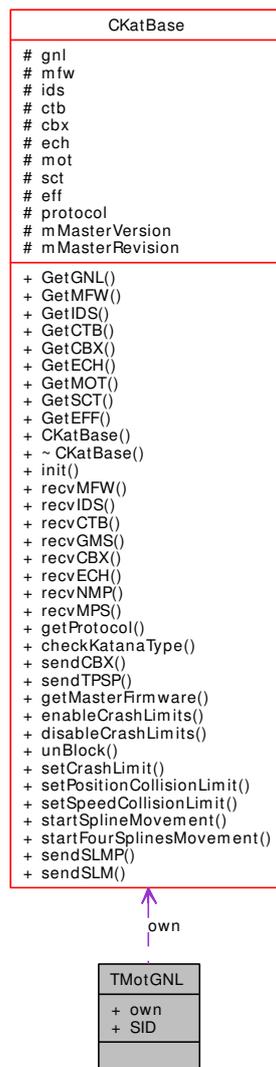
- [include/KNI/kmlMotBase.h](#)

11.75 TMotGNL Struct Reference

[GNL] motor generals

```
#include <kmlMotBase.h>
```

Collaboration diagram for TMotGNL:



Public Attributes

- [CKatBase * own](#)
parent robot
- [byte SID](#)
slave ID

11.75.1 Detailed Description

[GNL] motor generals

Definition at line 78 of file kmlMotBase.h.

11.75.2 Member Data Documentation

11.75.2.1 [CKatBase* TMotGNL::own](#)

parent robot

Definition at line 79 of file kmlMotBase.h.

11.75.2.2 [byte TMotGNL::SID](#)

slave ID

Definition at line 80 of file kmlMotBase.h.

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

- [include/KNI/kmlMotBase.h](#)

11.76 TMotInit Struct Reference

Initial motor parameters.

```
#include <kmlMotBase.h>
```

Public Attributes

- int [encoderOffset](#)
- int [encodersPerCycle](#)
- double [angleOffset](#)
- double [angleRange](#)
- int [rotationDirection](#)
- double [angleStop](#)

11.76.1 Detailed Description

Initial motor parameters.

Definition at line 198 of file kmlMotBase.h.

11.76.2 Member Data Documentation

11.76.2.1 int [TMotInit::encoderOffset](#)

Definition at line 199 of file kmlMotBase.h.

11.76.2.2 int [TMotInit::encodersPerCycle](#)

Definition at line 200 of file kmlMotBase.h.

11.76.2.3 double [TMotInit::angleOffset](#)

Definition at line 201 of file kmlMotBase.h.

11.76.2.4 double [TMotInit::angleRange](#)

Definition at line 202 of file kmlMotBase.h.

11.76.2.5 int [TMotInit::rotationDirection](#)

Definition at line 203 of file kmlMotBase.h.

11.76.2.6 double [TMotInit::angleStop](#)

Definition at line 206 of file kmlMotBase.h.

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

- [include/KNI/kmlMotBase.h](#)

11.77 TMotPVP Struct Reference

[PVP] position, velocity, pulse width modulation

```
#include <kmlMotBase.h>
```

Public Attributes

- [TMotStsFlg msf](#)
motor status flag
- short [pos](#)
position
- short [vel](#)
velocity
- [byte pwm](#)
pulse with modulation

11.77.1 Detailed Description

[PVP] position, velocity, pulse width modulation

Definition at line 161 of file kmlMotBase.h.

11.77.2 Member Data Documentation

11.77.2.1 [TMotStsFlg TMotPVP::msf](#)

motor status flag

Definition at line 162 of file kmlMotBase.h.

11.77.2.2 short [TMotPVP::pos](#)

position

Definition at line 163 of file kmlMotBase.h.

11.77.2.3 short [TMotPVP::vel](#)

velocity

Definition at line 164 of file kmlMotBase.h.

11.77.2.4 byte TMotPVP::pwm

pulse with modulation

Definition at line 165 of file kmlMotBase.h.

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

- [include/KNI/kmlMotBase.h](#)

11.78 TMotSCP Struct Reference

[SCP] static controller parameters

```
#include <kmlMotBase.h>
```

Public Attributes

- [byte maxppwm](#)
max. val for pos. voltage
- [byte maxnpwm](#)
max. val for neg. voltage; pos!
- [byte kP](#)
prop. factor of pos comp
- [byte kI](#)
not yet active
- [byte kD](#)
derivate factor of pos comp
- [byte kARW](#)
not yet active
- [byte kP_speed](#)
Proportional factor of the speed compensator.
- [byte kI_speed](#)
Integral factor of the speed compensator.
- [byte kD_speed](#)
Derivative factor of the speed compensator.
- [byte maxppwm_nmp](#)
Max. value for positive voltage (0 => 0%, +70 => 100%).
- [byte maxnpwm_nmp](#)
Max. value for negative voltage (0 => 0%, +70 => 100%).
- [byte kspeed_nmp](#)
Proportional factor of speed compensator.
- [byte kpos_nmp](#)
Proportional factor of position compensator.
- [byte kI_nmp](#)
Integral factor (I/kI) of control output added to the final control output.

- int [crash_limit_nmp](#)
Limit of error in position.
- int [crash_limit_lin_nmp](#)
Limit of error in position in linear movement.

11.78.1 Detailed Description

[SCP] static controller parameters

Definition at line 109 of file kmlMotBase.h.

11.78.2 Member Data Documentation

11.78.2.1 [byte TMotSCP::maxppwm](#)

max. val for pos. voltage

Definition at line 113 of file kmlMotBase.h.

11.78.2.2 [byte TMotSCP::maxnpwm](#)

max. val for neg. voltage; pos!

Definition at line 114 of file kmlMotBase.h.

11.78.2.3 [byte TMotSCP::kP](#)

prop. factor of pos comp

Definition at line 115 of file kmlMotBase.h.

11.78.2.4 [byte TMotSCP::kI](#)

not yet active

Definition at line 116 of file kmlMotBase.h.

11.78.2.5 [byte TMotSCP::kD](#)

derivate factor of pos comp

Definition at line 117 of file kmlMotBase.h.

11.78.2.6 [byte TMotSCP::kARW](#)

not yet active

Definition at line 118 of file kmlMotBase.h.

11.78.2.7 byte TMotSCP::kP_speed

Proportional factor of the speed compensator.

Definition at line 120 of file kmlMotBase.h.

11.78.2.8 byte TMotSCP::kI_speed

Integral factor of the speed compensator.

Definition at line 121 of file kmlMotBase.h.

11.78.2.9 byte TMotSCP::kD_speed

Derivative factor of the speed compensator.

Definition at line 122 of file kmlMotBase.h.

11.78.2.10 byte TMotSCP::maxppwm_nmp

Max. value for positive voltage (0 => 0%, +70 => 100%).

Definition at line 126 of file kmlMotBase.h.

11.78.2.11 byte TMotSCP::maxnpwm_nmp

Max. value for negative voltage (0 => 0%, +70 => 100%).

Definition at line 127 of file kmlMotBase.h.

11.78.2.12 byte TMotSCP::kspeed_nmp

Proportional factor of speed compensator.

Definition at line 128 of file kmlMotBase.h.

11.78.2.13 byte TMotSCP::kpos_nmp

Proportional factor of position compensator.

Definition at line 129 of file kmlMotBase.h.

11.78.2.14 byte TMotSCP::kI_nmp

Integral factor (1/kI) of control output added to the final control output.

Definition at line 130 of file kmlMotBase.h.

11.78.2.15 int TMotSCP::crash_limit_nmp

Limit of error in position.

Definition at line 131 of file kmlMotBase.h.

11.78.2.16 `int` [TMotSCP::crash_limit_lin_nmp](#)

Limit of error in position in linear movement.

Definition at line 132 of file `kmlMotBase.h`.

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

- [include/KNI/kmlMotBase.h](#)

11.79 TMotSFW Struct Reference

[SFW] slave firmware

```
#include <kmlMotBase.h>
```

Public Attributes

- [byte version](#)
firmware version number
- [byte subversion](#)
firmware subversion number
- [byte revision](#)
firmware revision number
- [byte type](#)
firmware type
- [byte subtype](#)
firmware subtype

11.79.1 Detailed Description

[SFW] slave firmware

Definition at line 85 of file kmlMotBase.h.

11.79.2 Member Data Documentation

11.79.2.1 [byte TMotSFW::version](#)

firmware version number

Definition at line 86 of file kmlMotBase.h.

11.79.2.2 [byte TMotSFW::subversion](#)

firmware subversion number

Definition at line 87 of file kmlMotBase.h.

11.79.2.3 [byte TMotSFW::revision](#)

firmware revision number

Definition at line 88 of file kmlMotBase.h.

11.79.2.4 byte TMotSFW::type

firmware type

Definition at line 89 of file kmlMotBase.h.

11.79.2.5 byte TMotSFW::subtype

firmware subtype

Definition at line 90 of file kmlMotBase.h.

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

- [include/KNI/kmlMotBase.h](#)

11.80 TMotTPS Struct Reference

[TPS] target position

```
#include <kmlMotBase.h>
```

Public Attributes

- [TMotCmdFlg mcfTPS](#)
motor command flag
- short [tarpos](#)
target position

11.80.1 Detailed Description

[TPS] target position

Definition at line 102 of file kmlMotBase.h.

11.80.2 Member Data Documentation

11.80.2.1 [TMotCmdFlg TMotTPS::mcfTPS](#)

motor command flag

Definition at line 103 of file kmlMotBase.h.

11.80.2.2 short [TMotTPS::tarpos](#)

target position

Definition at line 104 of file kmlMotBase.h.

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

- [include/KNI/kmlMotBase.h](#)

11.81 TPacket Struct Reference

Communication packet.

```
#include <cplSerial.h>
```

Public Attributes

- [byte send_sz](#)
send size of the packet
- [byte read_sz](#)
read size of the packet

11.81.1 Detailed Description

Communication packet.

Definition at line 63 of file cplSerial.h.

11.81.2 Member Data Documentation

11.81.2.1 [byte TPacket::send_sz](#)

send size of the packet

Definition at line 64 of file cplSerial.h.

11.81.2.2 [byte TPacket::read_sz](#)

read size of the packet

Definition at line 65 of file cplSerial.h.

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

- [include/KNI/cplSerial.h](#)

11.82 TPoint3D Struct Reference

```
#include <lmBase.h>
```

Public Attributes

- double [X](#)
- double [Y](#)
- double [Z](#)

11.82.1 Detailed Description

Definition at line 76 of file lmBase.h.

11.82.2 Member Data Documentation

11.82.2.1 double [TPoint3D::X](#)

Definition at line 77 of file lmBase.h.

11.82.2.2 double [TPoint3D::Y](#)

Definition at line 78 of file lmBase.h.

11.82.2.3 double [TPoint3D::Z](#)

Definition at line 79 of file lmBase.h.

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

- include/KNI_LM/lmBase.h

11.83 TPoint6D Struct Reference

[LMBLEND] Standard coordinates for a point in space

```
#include <lmBase.h>
```

Public Attributes

- double [X](#)
- double [Y](#)
- double [Z](#)
- double [Al](#)
- double [Be](#)
- double [Ga](#)

11.83.1 Detailed Description

[LMBLEND] Standard coordinates for a point in space

Definition at line 67 of file lmBase.h.

11.83.2 Member Data Documentation

11.83.2.1 double [TPoint6D::X](#)

Definition at line 68 of file lmBase.h.

11.83.2.2 double [TPoint6D::Y](#)

Definition at line 69 of file lmBase.h.

11.83.2.3 double [TPoint6D::Z](#)

Definition at line 70 of file lmBase.h.

11.83.2.4 double [TPoint6D::Al](#)

Definition at line 71 of file lmBase.h.

11.83.2.5 double [TPoint6D::Be](#)

Definition at line 72 of file lmBase.h.

11.83.2.6 double [TPoint6D::Ga](#)

Definition at line 73 of file lmBase.h.

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

- [include/KNI_LM/lmBase.h](#)

11.84 TSctDAT Struct Reference

[DAT] sensor data

```
#include <kmlSctBase.h>
```

Public Attributes

- short [cnt](#)
count of sensors
- short * [arr](#)
sensor data

11.84.1 Detailed Description

[DAT] sensor data

Definition at line 57 of file kmlSctBase.h.

11.84.2 Member Data Documentation

11.84.2.1 short [TSctDAT::cnt](#)

count of sensors

Definition at line 58 of file kmlSctBase.h.

11.84.2.2 short* [TSctDAT::arr](#)

sensor data

Definition at line 59 of file kmlSctBase.h.

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

- [include/KNI/kmlSctBase.h](#)

11.85 TSctDesc Struct Reference

sensor controller description (partly)

```
#include <kmlSctBase.h>
```

Public Attributes

- [byte ctrlID](#)
controller number (ID)
- [short sens_res](#)
resolution: 8/12 bit
- [short sens_count](#)
count of sensors

11.85.1 Detailed Description

sensor controller description (partly)

Definition at line 33 of file kmlSctBase.h.

11.85.2 Member Data Documentation

11.85.2.1 [byte TSctDesc::ctrlID](#)

controller number (ID)

Definition at line 34 of file kmlSctBase.h.

11.85.2.2 [short TSctDesc::sens_res](#)

resolution: 8/12 bit

Definition at line 35 of file kmlSctBase.h.

11.85.2.3 [short TSctDesc::sens_count](#)

count of sensors

Definition at line 36 of file kmlSctBase.h.

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

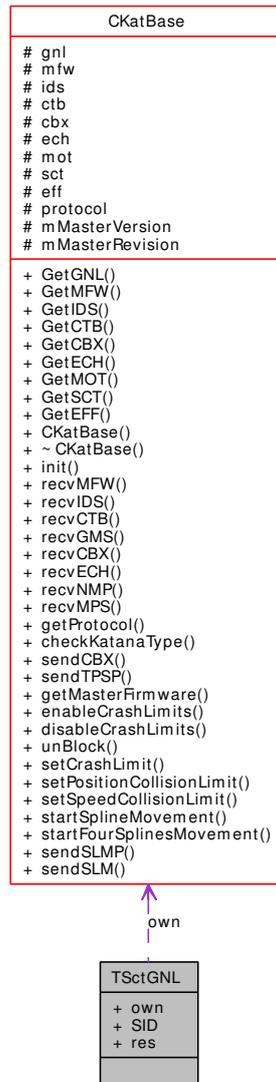
- [include/KNI/kmlSctBase.h](#)

11.86 TSctGNL Struct Reference

[GNL] controller generals

```
#include <kmlSctBase.h>
```

Collaboration diagram for TSctGNL:



Public Attributes

- [CKatBase * own](#)
parent robot
- [byte SID](#)
slave ID
- [short res](#)

resolution: 8/12 bit

11.86.1 Detailed Description

[GNL] controller generals

Definition at line 49 of file kmlSctBase.h.

11.86.2 Member Data Documentation

11.86.2.1 **CKatBase* TSctGNL::own**

parent robot

Definition at line 50 of file kmlSctBase.h.

11.86.2.2 **byte TSctGNL::SID**

slave ID

Definition at line 51 of file kmlSctBase.h.

11.86.2.3 **short TSctGNL::res**

resolution: 8/12 bit

Definition at line 52 of file kmlSctBase.h.

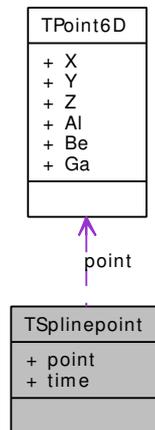
The documentation for this struct was generated from the following file:

- [include/KNI/kmlSctBase.h](#)

11.87 TSplinepoint Struct Reference

```
#include <lmBase.h>
```

Collaboration diagram for TSplinepoint:



Public Attributes

- [TPoint6D point](#)
- double [time](#)

11.87.1 Detailed Description

Definition at line 99 of file lmBase.h.

11.87.2 Member Data Documentation

11.87.2.1 [TPoint6D TSplinepoint::point](#)

Definition at line 100 of file lmBase.h.

11.87.2.2 [double TSplinepoint::time](#)

Definition at line 101 of file lmBase.h.

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

- [include/KNI_LM/lmBase.h](#)

11.88 KNI_MHF::unary_deg2rad< _T > Struct Template Reference

a function-object version of rad2deg

```
#include <MathHelperFunctions.h>
```

Public Member Functions

- [_T operator\(\)](#) (const _T a)

11.88.1 Detailed Description

```
template<typename _T> struct KNI_MHF::unary_deg2rad< _T >
```

a function-object version of rad2deg

Definition at line 121 of file MathHelperFunctions.h.

11.88.2 Member Function Documentation

11.88.2.1 `template<typename _T> _T KNI_MHF::unary_deg2rad< _T >::operator() (const _T a) [inline]`

Definition at line 122 of file MathHelperFunctions.h.

References [KNI_MHF::deg2rad\(\)](#).

Here is the call graph for this function:



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

- [include/common/MathHelperFunctions.h](#)

11.89 KNI_MHF::unary_precalc_cos< _T > Struct Template Reference

See also:

[unary_precalc_sin](#)

```
#include <MathHelperFunctions.h>
```

Public Member Functions

- [_T operator\(\) \(_T x\)](#)

11.89.1 Detailed Description

```
template<typename _T> struct KNI_MHF::unary_precalc_cos< _T >
```

See also:

[unary_precalc_sin](#)

Definition at line 53 of file MathHelperFunctions.h.

11.89.2 Member Function Documentation

11.89.2.1 `template<typename _T> _T KNI_MHF::unary_precalc_cos< _T >::operator() (_T x)`
[inline]

Definition at line 54 of file MathHelperFunctions.h.

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

- [include/common/MathHelperFunctions.h](#)

11.90 KNI_MHF::unary_precalc_sin< _T > Struct Template Reference

function-object which calculates sinus for n-elements of a container if used together with a STL algorithm

```
#include <MathHelperFunctions.h>
```

Public Member Functions

- [_T operator\(\)](#) (_T &x)

11.90.1 Detailed Description

```
template<typename _T> struct KNI_MHF::unary_precalc_sin< _T >
```

function-object which calculates sinus for n-elements of a container if used together with a STL algorithm

Definition at line 44 of file MathHelperFunctions.h.

11.90.2 Member Function Documentation

11.90.2.1 `template<typename _T> _T KNI_MHF::unary_precalc_sin< _T >::operator() (_T &x) [inline]`

Definition at line 45 of file MathHelperFunctions.h.

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

- `include/common/MathHelperFunctions.h`

11.91 KNI_MHF::unary_rad2deg< _T > Struct Template Reference

a function-object version of rad2deg

```
#include <MathHelperFunctions.h>
```

Public Member Functions

- [_T operator\(\)](#) (const _T a)

11.91.1 Detailed Description

`template<typename _T> struct KNI_MHF::unary_rad2deg< _T >`

a function-object version of rad2deg

Definition at line 107 of file MathHelperFunctions.h.

11.91.2 Member Function Documentation

11.91.2.1 `template<typename _T> _T KNI_MHF::unary_rad2deg< _T >::operator() (const _T a) [inline]`

Definition at line 108 of file MathHelperFunctions.h.

References `KNI_MHF::rad2deg()`.

Here is the call graph for this function:



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

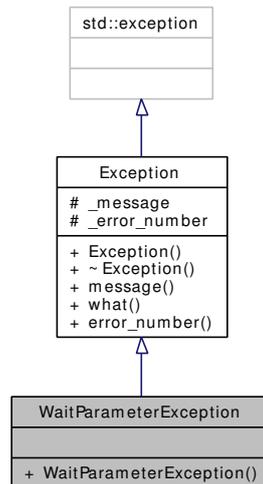
- [include/common/MathHelperFunctions.h](#)

11.92 WaitParameterException Class Reference

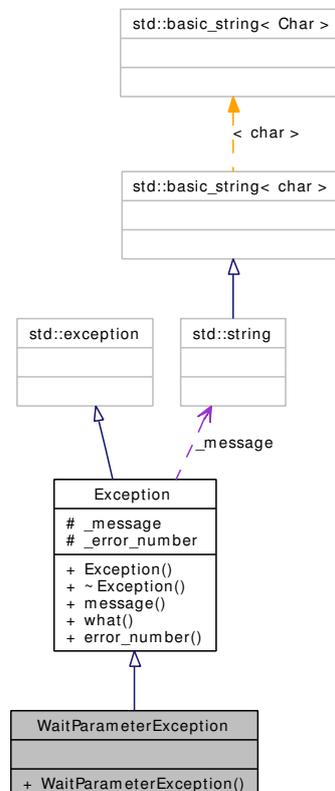
Wait parameter set to false.

```
#include <lmBase.h>
```

Inheritance diagram for WaitParameterException:



Collaboration diagram for WaitParameterException:



Public Member Functions

- [WaitParameterException \(\) throw \(\)](#)

11.92.1 Detailed Description

Wait parameter set to false.

Note:

error_number = -71

Definition at line 137 of file lmBase.h.

11.92.2 Constructor & Destructor Documentation

11.92.2.1 WaitParameterException::WaitParameterException () throw () [inline]

Definition at line 139 of file lmBase.h.

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

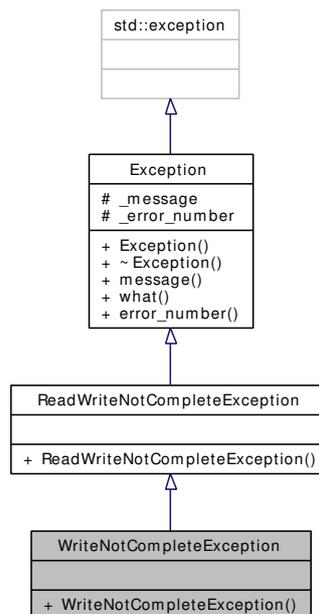
- [include/KNI_LM/lmBase.h](#)

11.93 WriteNotCompleteException Class Reference

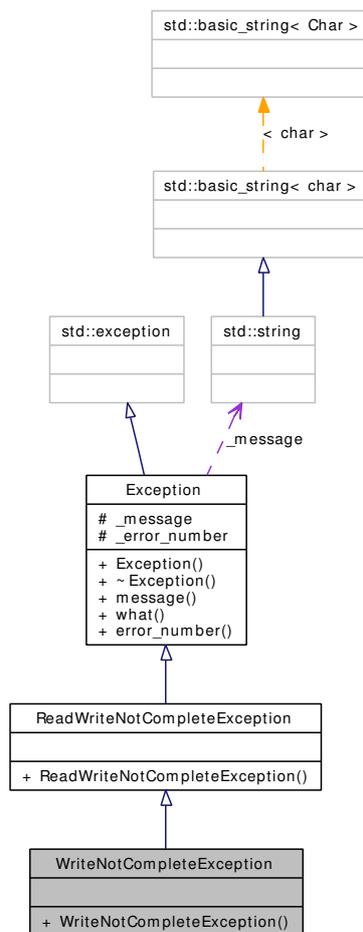
Not all bytes could be written to the serial communication device.

```
#include <cdlCOMExceptions.h>
```

Inheritance diagram for WriteNotCompleteException:



Collaboration diagram for WriteNotCompleteException:



Public Member Functions

- [WriteNotCompleteException](#) (const std::string &port) throw ()

11.93.1 Detailed Description

Not all bytes could be written to the serial communication device.

Note:

error_number=-15

Definition at line 103 of file cdICOMExceptions.h.

11.93.2 Constructor & Destructor Documentation

11.93.2.1 WriteNotCompleteException::WriteNotCompleteException (const std::string &port) throw () [inline]

Definition at line 105 of file cdICOMExceptions.h.

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

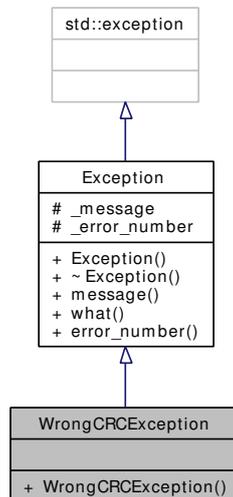
- [include/KNI/cdlCOMExceptions.h](#)

11.94 WrongCRCEXception Class Reference

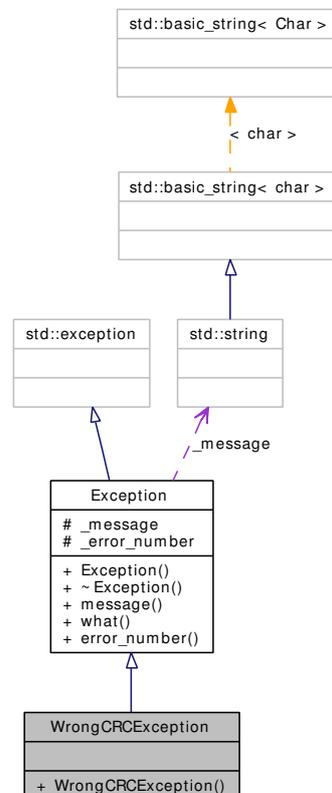
CRC check for the answer package failed.

```
#include <cplSerial.h>
```

Inheritance diagram for WrongCRCEXception:



Collaboration diagram for WrongCRCEXception:



Public Member Functions

- [WrongCRCEXception](#) () throw ()

11.94.1 Detailed Description

CRC check for the answer package failed.

Definition at line 44 of file `cplSerial.h`.

11.94.2 Constructor & Destructor Documentation

11.94.2.1 `WrongCRCEXception::WrongCRCEXception () throw ()` `[inline]`

Definition at line 46 of file `cplSerial.h`.

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

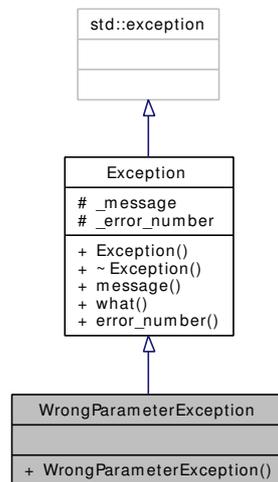
- `include/KNI/cplSerial.h`

11.95 WrongParameterException Class Reference

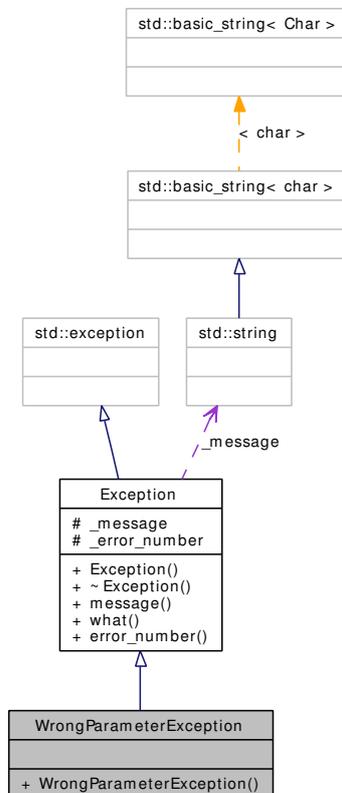
The given parameter was wrong.

```
#include <kmlCommon.h>
```

Inheritance diagram for WrongParameterException:



Collaboration diagram for WrongParameterException:



Public Member Functions

- [WrongParameterException](#) (const std::string ¶) throw ()

11.95.1 Detailed Description

The given parameter was wrong.

Note:

```
error_number=-34
```

Definition at line 62 of file kmlCommon.h.

11.95.2 Constructor & Destructor Documentation

11.95.2.1 WrongParameterException::WrongParameterException (const std::string & para) throw () [inline]

Definition at line 64 of file kmlCommon.h.

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

- [include/KNI/kmlCommon.h](#)

12.1.1 Define Documentation

12.1.1.1 #define DLLDIR

Definition at line 30 of file dllexport.h.

12.1.1.2 #define DLLDIR_IK

Definition at line 31 of file dllexport.h.

12.1.1.3 #define DLLDIR_LM

Definition at line 32 of file dllexport.h.

12.2 include/common/exception.h File Reference

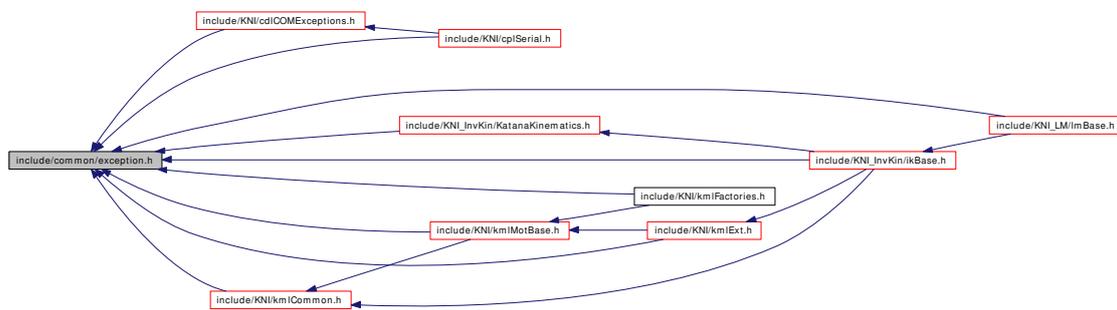
```
#include <string>
```

```
#include "dllexport.h"
```

Include dependency graph for exception.h:



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



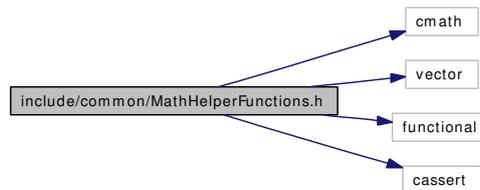
Classes

- struct [Context](#)
- class [Exception](#)

12.3 include/common/MathHelperFunctions.h File Reference

```
#include <cmath>
#include <vector>
#include <functional>
#include <cassert>
```

Include dependency graph for MathHelperFunctions.h:



Namespaces

- namespace [KNI_MHF](#)

Classes

- struct [KNI_MHF::unary_precalc_sin< _T >](#)
function-object which calculates sinus for n-elements of a container if used together with a STL algorithm
- struct [KNI_MHF::unary_precalc_cos< _T >](#)
See also:
[unary_precalc_sin](#)
- struct [KNI_MHF::unary_rad2deg< _T >](#)
a function-object version of rad2deg
- struct [KNI_MHF::unary_deg2rad< _T >](#)
a function-object version of rad2deg

Defines

- #define [M_PI](#) 3.14159265358979323846

Functions

- template<typename _T> short [KNI_MHF::sign](#) (_T x)
- template<typename _T> _T [KNI_MHF::atan1](#) (_T in1, _T in2)
- template<typename _T> _T [KNI_MHF::acotan](#) (const _T in)
- template<typename _T> _T [KNI_MHF::atan0](#) (const _T in1, const _T in2)
- template<typename _T> _T [KNI_MHF::pow2](#) (const _T in)

- `template<typename _T> _T KNI_MHF::rad2deg (const _T a)`
conversion from radian to degree
- `template<typename _T> _T KNI_MHF::deg2rad (const _T a)`
conversion from degree to radian
- `template<typename _T> _T KNI_MHF::anglereduce (const _T a)`
- `template<typename _angleT, typename _encT> _encT KNI_MHF::rad2enc (_angleT const &angle, _angleT const &angleOffset, _encT const &enc, _encT const &encOffset, _encT const &rotDir)`
converts absolute angles in radian to encoders.
- `template<typename _angleT, typename _encT> _angleT KNI_MHF::enc2rad (_encT const &enc, _angleT const &angleOffset, _encT const &enc, _encT const &encOffset, _encT const &rotDir)`
converts encoders to absolute angles in radian
- `double KNI_MHF::findFirstEqualAngle (double cosValue, double sinValue, double tolerance)`
Find the first equal angle.

12.3.1 Define Documentation

12.3.1.1 #define M_PI 3.14159265358979323846

Definition at line 21 of file MathHelperFunctions.h.

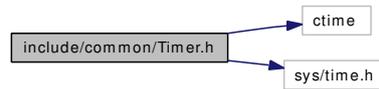
Referenced by `KNI_MHF::acotan()`, `KNI_MHF::anglereduce()`, `KNI_MHF::atan0()`, `KNI_MHF::atan1()`, `KNI_MHF::deg2rad()`, `KNI_MHF::enc2rad()`, `KNI_MHF::findFirstEqualAngle()`, `KNI_MHF::rad2deg()`, and `KNI_MHF::rad2enc()`.

12.4 include/common/Timer.h File Reference

```
#include <ctime>
```

```
#include <sys/time.h>
```

Include dependency graph for Timer.h:



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



Namespaces

- namespace [KNI](#)

Classes

- class [KNI::Timer](#)
Provides a stop-watch-like class with a resolution of milliseconds.

Functions

- void [KNI::sleep](#) (long time)
This functions shields the platform specific implementation of the sleep function.

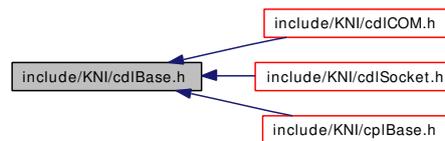
12.5 include/KNI/cdlBase.h File Reference

```
#include "common/dllexport.h"
```

Include dependency graph for cdlBase.h:



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



Classes

- class [CCdlBase](#)
Abstract base class for devices.

Defines

- #define [BYTE_DECLARED](#)

Typedefs

- typedef unsigned char [byte](#)
type specification (8 bit)

12.5.1 Define Documentation

12.5.1.1 #define [BYTE_DECLARED](#)

Definition at line 28 of file cdlBase.h.

12.5.2 Typedef Documentation

12.5.2.1 typedef unsigned char [byte](#)

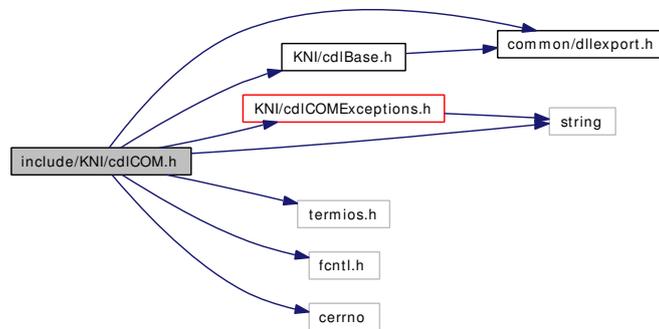
type specification (8 bit)

Definition at line 29 of file cdlBase.h.

12.6 include/KNI/cdlCOM.h File Reference

```
#include "common/dllexport.h"
#include "KNI/cdlBase.h"
#include "KNI/cdlCOMExceptions.h"
#include <string>
#include <termios.h>
#include <fcntl.h>
#include <cerrno>
```

Include dependency graph for cdlCOM.h:



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



Classes

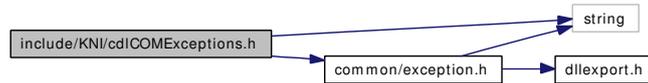
- struct [TCdlCOMDesc](#)
This struct stores the attributes for a serial port device.
- class [CCdlCOM](#)
Encapsulates the serial port device.

12.7 include/KNI/cdlCOMExceptions.h File Reference

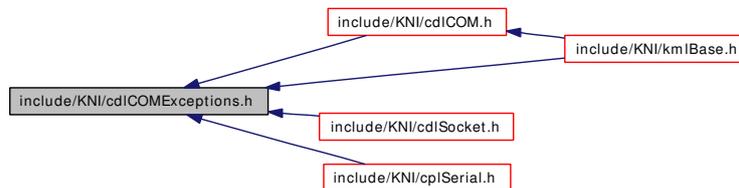
```
#include "common/exception.h"
```

```
#include <string>
```

Include dependency graph for cdlCOMExceptions.h:



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



Classes

- class [CannotOpenPortException](#)
Failed to open the serial communication device.
- class [CannotGetSetPortAttributesException](#)
Could not set or get the attributes for the given serial communication device.
- class [PortNotOpenException](#)
The port was not open.
- class [DeviceReadException](#)
Reading from the serial communication device failed.
- class [DeviceWriteException](#)
Writing to the serial communication device failed.
- class [ReadWriteNotCompleteException](#)
This exception is the base for the WriteNotComplete and ReadNotCompleteException.
- class [WriteNotCompleteException](#)
Not all bytes could be written to the serial communication device.
- class [ReadNotCompleteException](#)
The Katana didn't answer correctly within the given timeout.
- class [ErrorException](#)
The Katana returned an error string.

Enumerations

- enum {
 ERR_FAILED = -1, ERR_INVALID_ARGUMENT = -2, ERR_STATE_MISMATCH = -3, ERR_-
 TYPE_MISMATCH = -4,
 ERR_RANGE_MISMATCH = -5, ERR_AXIS_HEARTBEAT = -6, ERR_AXIS_OPERATIONAL
 = -7, ERR_AXIS_MOVE = -8,
 ERR_AXIS_MOVE_POLY = -9, ERR_AXIS_COLLISION = -10, ERR_AXIS_ANY = -11, ERR_-
 CRC = -12,
 ERR_PERIPHERAL = -13, ERR_MESSAGE = 192, ERR_MESSAGE_STRING = 193 }

Error codes in error handling strings.

12.7.1 Enumeration Type Documentation

12.7.1.1 anonymous enum

Error codes in error handling strings.

Enumerator:

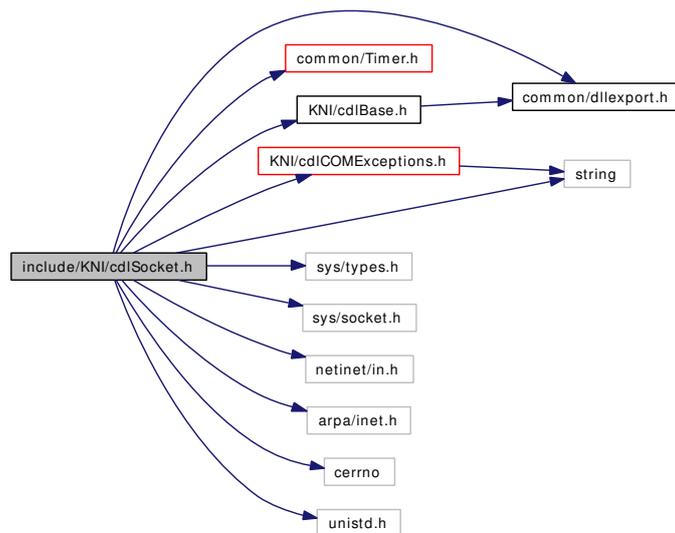
ERR_FAILED
ERR_INVALID_ARGUMENT
ERR_STATE_MISMATCH
ERR_TYPE_MISMATCH
ERR_RANGE_MISMATCH
ERR_AXIS_HEARTBEAT
ERR_AXIS_OPERATIONAL
ERR_AXIS_MOVE
ERR_AXIS_MOVE_POLY
ERR_AXIS_COLLISION
ERR_AXIS_ANY
ERR_CRC
ERR_PERIPHERAL
ERR_MESSAGE
ERR_MESSAGE_STRING

Definition at line 20 of file cdIComExceptions.h.

12.8 include/KNI/cdlSocket.h File Reference

```
#include "common/dllexport.h"
#include "common/Timer.h"
#include "KNI/cdlBase.h"
#include "KNI/cdlCOMExceptions.h"
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <cerrno>
#include <unistd.h>
#include <string>
```

Include dependency graph for cdlSocket.h:



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



Classes

- class [CCdlSocket](#)

Encapsulates the socket communication device.

12.9 include/KNI/cplBase.h File Reference

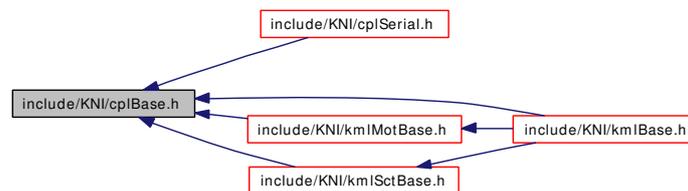
```
#include "common/dllexport.h"
```

```
#include "KNI/cdlBase.h"
```

Include dependency graph for cplBase.h:



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



Classes

- class [CCplBase](#)
Abstract base class for protocol definiton.

Defines

- #define [BYTE_DECLARED](#)

Typedefs

- typedef unsigned char [byte](#)
type specification (8 bit)

12.9.1 Define Documentation

12.9.1.1 #define BYTE_DECLARED

Definition at line 32 of file cplBase.h.

12.9.2 Typedef Documentation

12.9.2.1 typedef unsigned char byte

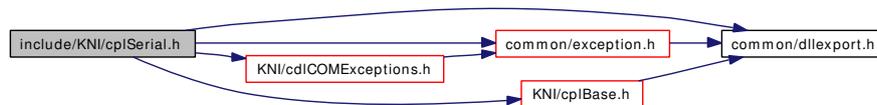
type specification (8 bit)

Definition at line 33 of file cplBase.h.

12.10 include/KNI/cplSerial.h File Reference

```
#include "common/dllexport.h"
#include "common/exception.h"
#include "KNI/cplBase.h"
#include "KNI/cdlCOMExceptions.h"
```

Include dependency graph for cplSerial.h:



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



Classes

- class [WrongCRCException](#)
CRC check for the answer package failed.
- struct [THeader](#)
Header of a communication packet.
- struct [TPacket](#)
Communication packet.
- class [CCplSerial](#)
Base class of two different serial protocols.
- class [CCplSerialCRC](#)
Implement the Serial-Zero protocol Initializing functionCommunication functionImplement the Serial-CRC protocol.

Defines

- #define [NUMBER_OF_RETRIES_SEND](#) 3
- #define [NUMBER_OF_RETRIES_RECV](#) 3

Variables

- const int [KATANA_ERROR_FLAG](#) = 192
defines the error flag number

12.10.1 Define Documentation

12.10.1.1 `#define` NUMBER_OF_RETRIES_RECV 3

Definition at line 32 of file cplSerial.h.

12.10.1.2 `#define` NUMBER_OF_RETRIES_SEND 3

Definition at line 31 of file cplSerial.h.

12.10.2 Variable Documentation

12.10.2.1 const int [KATANA_ERROR_FLAG](#) = 192

defines the error flag number

Definition at line 36 of file cplSerial.h.

12.11 include/KNI/CRC.h File Reference

Defines

- #define [uint8](#) unsigned char
unsigned 8 bit
- #define [uint16](#) unsigned short
unsigned 16 bit

Functions

- uint16 [CRC_CHECKSUM](#) (uint8 *data, uint8 size_of_BYTE)

12.11.1 Define Documentation

12.11.1.1 #define uint16 unsigned short

unsigned 16 bit

Definition at line 28 of file CRC.h.

12.11.1.2 #define uint8 unsigned char

unsigned 8 bit

Definition at line 27 of file CRC.h.

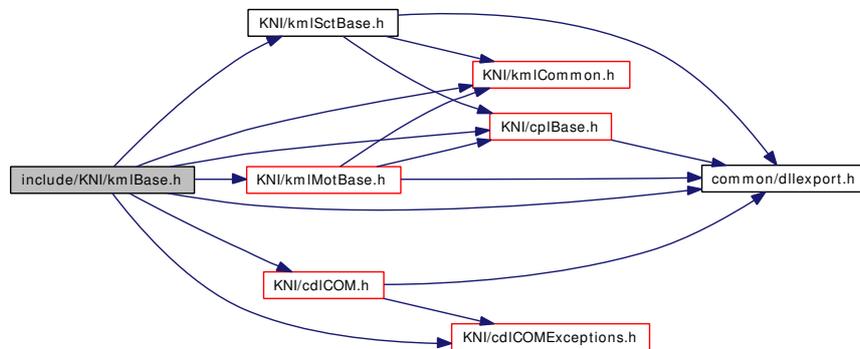
12.11.2 Function Documentation

12.11.2.1 uint16 CRC_CHECKSUM (uint8 * data, uint8 size_of_BYTE)

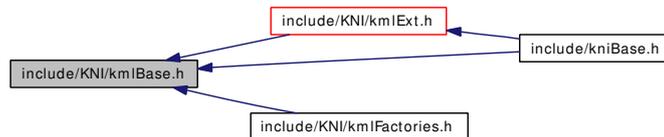
12.12 include/KNI/kmlBase.h File Reference

```
#include "common/dllexport.h"
#include "KNI/cplBase.h"
#include "KNI/kmlCommon.h"
#include "KNI/kmlMotBase.h"
#include "KNI/kmlSctBase.h"
#include "KNI/cdlCOM.h"
#include "KNI/cdlCOMExceptions.h"
```

Include dependency graph for kmlBase.h:



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



Classes

- struct [TKatGNL](#)
[GNL] general robot attributes
- struct [TKatMFW](#)
[MFW] master firmware version/revision number
- struct [TKatIDS](#)
[IDS] identification string
- struct [TKatCTB](#)
[CTB] command table defined in the firmware
- struct [TKatCBX](#)

[CBX] connector box

- struct [TKatECH](#)

[ECH] echo

- struct [TKatEFF](#)

Inverse Kinematics structure of the endeffektor.

- class [CKatBase](#)

Base Katana class.

Defines

- #define [K400_OLD_PROTOCOL_THRESHOLD](#) 1

The old protocol is only supported up to K400 version 0.x.x.

- #define [BYTE_DECLARED](#)

- #define [TM_ENDLESS](#) -1

timeout symbol for 'endless' waiting

Typedefs

- typedef unsigned char [byte](#)

type specification (8 bit)

12.12.1 Define Documentation

12.12.1.1 #define [BYTE_DECLARED](#)

Definition at line 45 of file kmlBase.h.

12.12.1.2 #define [K400_OLD_PROTOCOL_THRESHOLD](#) 1

The old protocol is only supported up to K400 version 0.x.x.

Definition at line 42 of file kmlBase.h.

12.12.1.3 #define [TM_ENDLESS](#) -1

timeout symbol for 'endless' waiting

Definition at line 51 of file kmlBase.h.

12.12.2 Typedef Documentation

12.12.2.1 typedef unsigned char `byte`

type specification (8 bit)

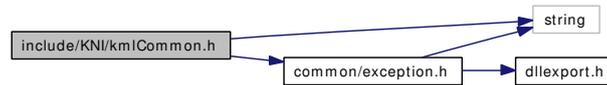
Definition at line 46 of file kmlBase.h.

12.13 include/KNI/kmlCommon.h File Reference

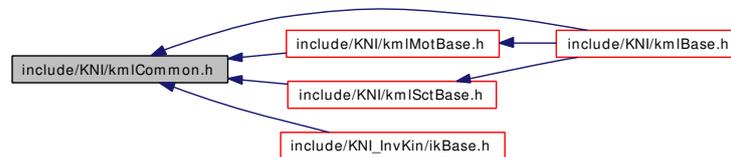
```
#include "common/exception.h"
```

```
#include <string>
```

Include dependency graph for kmlCommon.h:



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



Classes

- class [SlaveErrorException](#)
Slave error occurred.
- class [ParameterReadingException](#)
There was an error while reading a parameter from the robot.
- class [ParameterWritingException](#)
The data you wanted to send to the robot was invalid.
- class [WrongParameterException](#)
The given parameter was wrong.
- class [MotorOutOfRangeException](#)
The encoders for the given motor were out of range.
- class [MotorTimeoutException](#)
The timeout elapsed for the given motor and target position.
- class [MotorCrashException](#)
The requested motor crashed during the movement.

Defines

- `#define` [TM_ENDLESS](#) -1
timeout symbol for 'endless' waiting

- #define [BYTE_DECLARED](#)

Typedefs

- typedef unsigned char [byte](#)
type specification (8 bit)

12.13.1 Define Documentation

12.13.1.1 #define [BYTE_DECLARED](#)

Definition at line 22 of file kmlCommon.h.

12.13.1.2 #define [TM_ENDLESS](#) -1

timeout symbol for 'endless' waiting

Definition at line 19 of file kmlCommon.h.

12.13.2 Typedef Documentation

12.13.2.1 typedef unsigned char [byte](#)

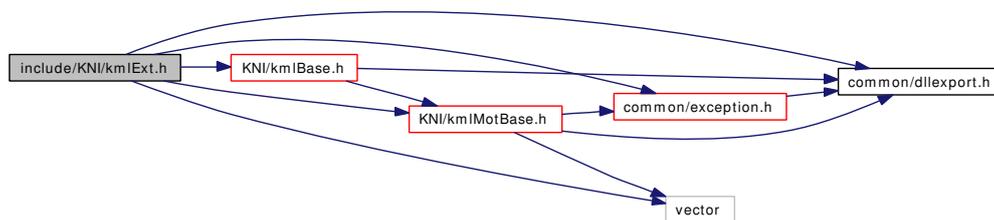
type specification (8 bit)

Definition at line 23 of file kmlCommon.h.

12.14 include/KNI/kmlExt.h File Reference

```
#include "common/dllexport.h"
#include "common/exception.h"
#include "KNI/kmlBase.h"
#include "KNI/kmlMotBase.h"
#include <vector>
```

Include dependency graph for kmlExt.h:



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



Namespaces

- namespace [KNI](#)

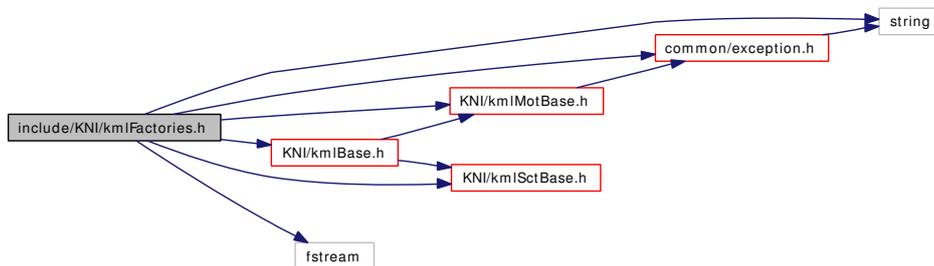
Classes

- class [ConfigFileOpenException](#)
Accessing the given configuration file failed (may be: access denied or wrong path).
- class [CKatana](#)
Extended Katana class with additional functions.

12.15 include/KNI/kmlFactories.h File Reference

```
#include "common/exception.h"
#include "KNI/kmlBase.h"
#include "KNI/kmlMotBase.h"
#include "KNI/kmlSctBase.h"
#include <string>
#include <fstream>
```

Include dependency graph for kmlFactories.h:



Namespaces

- namespace [KNI](#)

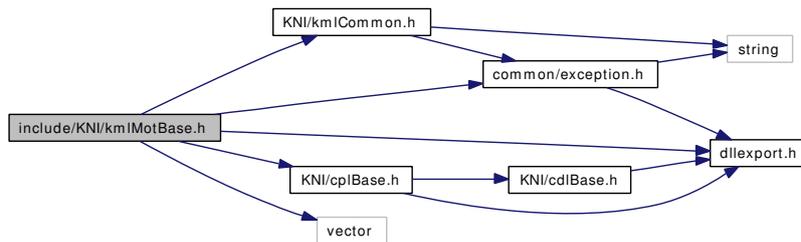
Classes

- class [ConfigFileStateException](#)
The state of the configuration file wasn't "good".
- class [ConfigFileSectionNotFoundException](#)
The requested section could not be found.
- class [ConfigFileSubsectionNotFoundException](#)
The requested subsection could not be found.
- class [ConfigFileEntryNotFoundException](#)
The requested entry could not be found.
- class [ConfigFileSyntaxErrorException](#)
There was a syntax error in the configuration file.
- class [KNI::kmlFactory](#)
This class is for internal use only It may change at any time It shields the configuration file parsing.

12.16 include/KNI/kmIMotBase.h File Reference

```
#include "common/exception.h"
#include "common/dllexport.h"
#include "KNI/kmICommon.h"
#include "KNI/cplBase.h"
#include <vector>
```

Include dependency graph for kmIMotBase.h:



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



Classes

- struct [TMotDesc](#)
motor description (partly)
- struct [TKatMOT](#)
[MOT] every motor's attributes
- struct [TMotGNL](#)
[GNL] motor generals
- struct [TMotSFW](#)
[SFW] slave firmware
- struct [TMotAPS](#)
[APS] actual position
- struct [TMotTPS](#)
[TPS] target position
- struct [TMotSCP](#)

[SCP] static controller parameters

- struct [TMotDYL](#)
[DYL] dynamic limits
- struct [TMotPVP](#)
[PVP] position, velocity, pulse width modulation
- struct [TMotENL](#)
[ENL] limits in encoder values (INTERNAL STRUCTURE!)
- struct [TMotCLB](#)
Calibration structure for single motors.
- struct [TMotInit](#)
Initial motor parameters.
- class [CMotBase](#)
Motor class.

Enumerations

- enum [TMotCmdFlg](#) { [MCF_OFF](#) = 0, [MCF_CALIB](#) = 4, [MCF_FREEZE](#) = 8, [MCF_ON](#) = 24 }
command flags
- enum [TMotStsFlg](#) {
[MSF_MECHSTOP](#) = 1, [MSF_MAXPOS](#) = 2, [MSF_MINPOS](#) = 4, [MSF_DESPOS](#) = 8,
[MSF_NORMOPSTAT](#) = 16, [MSF_MOTCRASHED](#) = 40, [MSF_NLINMOV](#) = 88, [MSF_LINMOV](#)
= 152,
[MSF_NOTVALID](#) = 128 }
status flags
- enum [TSearchDir](#) { [DIR_POSITIVE](#), [DIR_NEGATIVE](#) }

12.16.1 Enumeration Type Documentation

12.16.1.1 enum [TMotCmdFlg](#)

command flags

Enumerator:

MCF_OFF set the motor off
MCF_CALIB calibrate
MCF_FREEZE freeze the motor
MCF_ON set the motor on

Definition at line 48 of file kmlMotBase.h.

12.16.1.2 enum [TMotStsFlg](#)

status flags

Enumerator:

MSF_MECHSTOP mechanical stopper reached

MSF_MAXPOS max. position was reached

MSF_MINPOS min. position was reached

MSF_DESPOS in desired position

MSF_NORMOPSTAT trying to follow target

MSF_MOTCRASHED motor has crashed

MSF_NLINMOV non-linear movement ended

MSF_LINMOV linear movement ended

MSF_NOTVALID motor data not valid

Definition at line 57 of file kmlMotBase.h.

12.16.1.3 enum [TSearchDir](#)

Enumerator:

DIR_POSITIVE search direction for the meachanical stopper

DIR_NEGATIVE

Definition at line 68 of file kmlMotBase.h.

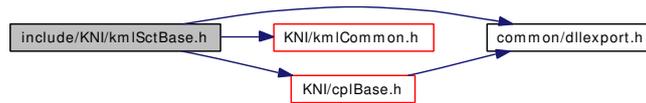
12.17 include/KNI/kmlSctBase.h File Reference

```
#include "common/dllexport.h"
```

```
#include "KNI/kmlCommon.h"
```

```
#include "KNI/cplBase.h"
```

Include dependency graph for kmlSctBase.h:



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



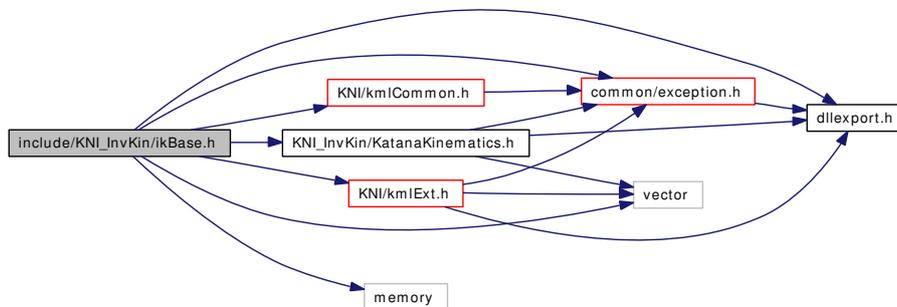
Classes

- struct [TSctDesc](#)
sensor controller description (partly)
- struct [TKatSCT](#)
[SCT] every sens ctrl's attributes
- struct [TSctGNL](#)
[GNL] controller generals
- struct [TSctDAT](#)
[DAT] sensor data
- class [CSctBase](#)
Sensor Controller class.

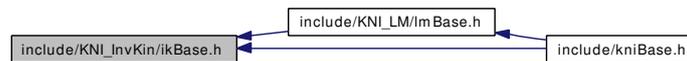
12.18 include/KNI_InvKin/ikBase.h File Reference

```
#include "common/exception.h"
#include "common/dllexport.h"
#include "KNI/kmlExt.h"
#include "KNI/kmlCommon.h"
#include "KNI_InvKin/KatanaKinematics.h"
#include <vector>
#include <memory>
```

Include dependency graph for ikBase.h:



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



Classes

- class [CikBase](#)

Defines

- #define [TM_ENDLESS](#) -1
timeout symbol for 'endless' waiting

12.18.1 Define Documentation

12.18.1.1 #define TM_ENDLESS -1

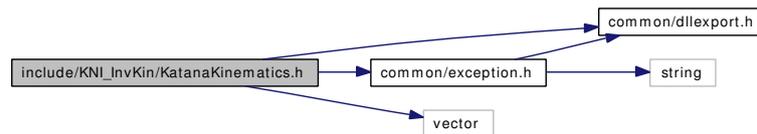
timeout symbol for 'endless' waiting

Definition at line 40 of file ikBase.h.

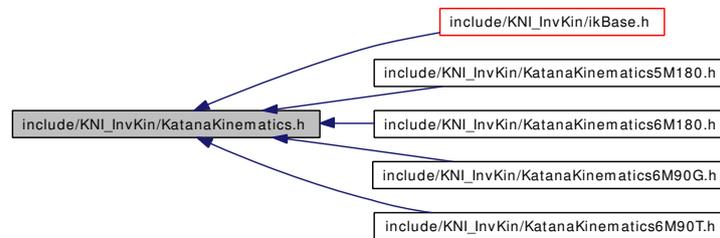
12.19 include/KNI_InvKin/KatanaKinematics.h File Reference

```
#include "common/dllexport.h"
#include "common/exception.h"
#include <vector>
```

Include dependency graph for KatanaKinematics.h:



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



Namespaces

- namespace [KNI](#)

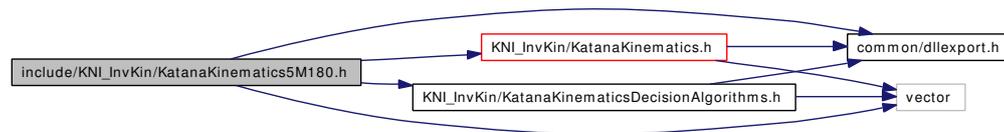
Classes

- class [KNI::NoSolutionException](#)
No solution found for the given cartesian coordinates.
- struct [KNI::KinematicParameters](#)
To pass different parameters for the kinematic implementations.
- class [KNI::KatanaKinematics](#)
The base class for all kinematic implementations.

12.20 include/KNI_InvKin/KatanaKinematics5M180.h File Reference

```
#include "common/dllexport.h"
#include "KNI_InvKin/KatanaKinematics.h"
#include "KNI_InvKin/KatanaKinematicsDecisionAlgorithms.h"
#include <vector>
```

Include dependency graph for KatanaKinematics5M180.h:



Namespaces

- namespace [KNI](#)

Classes

- class [KNI::KatanaKinematics5M180](#)

Author:

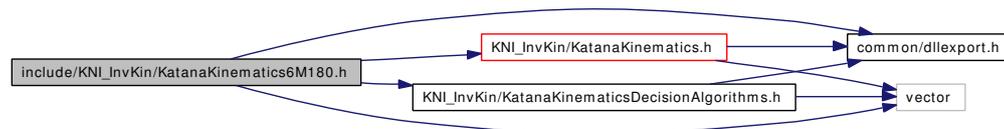
Tiziano Mueller <tiziano.mueller@neuronics.ch>

- struct [KNI::KatanaKinematics5M180::position](#)
- struct [KNI::KatanaKinematics5M180::angles_calc](#)

12.21 include/KNI_InvKin/KatanaKinematics6M180.h File Reference

```
#include "common/dllexport.h"  
#include "KNI_InvKin/KatanaKinematics.h"  
#include "KNI_InvKin/KatanaKinematicsDecisionAlgorithms.h"  
#include <vector>
```

Include dependency graph for KatanaKinematics6M180.h:



Namespaces

- namespace [KNI](#)

Classes

- class [KNI::KatanaKinematics6M180](#)

Author:

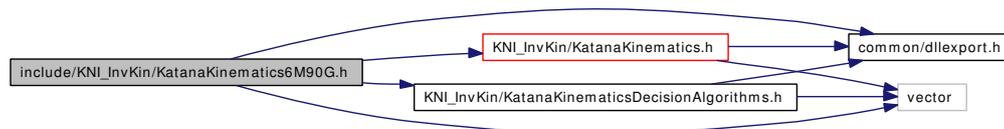
Tiziano Mueller <tiziano.mueller@neuronics.ch>

- struct [KNI::KatanaKinematics6M180::position](#)
- struct [KNI::KatanaKinematics6M180::angles_calc](#)

12.22 include/KNI_InvKin/KatanaKinematics6M90G.h File Reference

```
#include "common/dllexport.h"  
#include "KNI_InvKin/KatanaKinematics.h"  
#include "KNI_InvKin/KatanaKinematicsDecisionAlgorithms.h"  
#include <vector>
```

Include dependency graph for KatanaKinematics6M90G.h:



Namespaces

- namespace [KNI](#)

Classes

- class [KNI::KatanaKinematics6M90G](#)

Author:

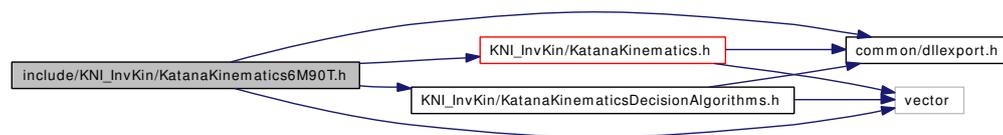
Tiziano Mueller <tiziano.mueller@neuronics.ch>

- struct [KNI::KatanaKinematics6M90G::position](#)
- struct [KNI::KatanaKinematics6M90G::angles_calc](#)

12.23 include/KNI_InvKin/KatanaKinematics6M90T.h File Reference

```
#include "common/dllexport.h"  
#include "KNI_InvKin/KatanaKinematics.h"  
#include "KNI_InvKin/KatanaKinematicsDecisionAlgorithms.h"  
#include <vector>
```

Include dependency graph for KatanaKinematics6M90T.h:



Namespaces

- namespace [KNI](#)

Classes

- class [KNI::KatanaKinematics6M90T](#)

Author:

Tiziano Mueller <tiziano.mueller@neuronics.ch>

- struct [KNI::KatanaKinematics6M90T::position](#)
- struct [KNI::KatanaKinematics6M90T::angles_calc](#)

12.24 include/KNI_InvKin/KatanaKinematicsDecision- Algorithms.h File Reference

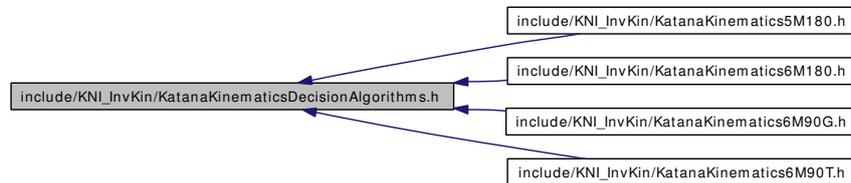
```
#include "common/dllexport.h"
```

```
#include <vector>
```

Include dependency graph for KatanaKinematicsDecisionAlgorithms.h:



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



Namespaces

- namespace [KNI](#)

Classes

- struct [KNI::KinematicsDefaultEncMinAlgorithm](#)

12.25 include/KNI_LM/lmBase.h File Reference

```
#include "KNI_InvKin/ikBase.h"
#include "common/exception.h"
#include <vector>
```

Include dependency graph for lmBase.h:



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



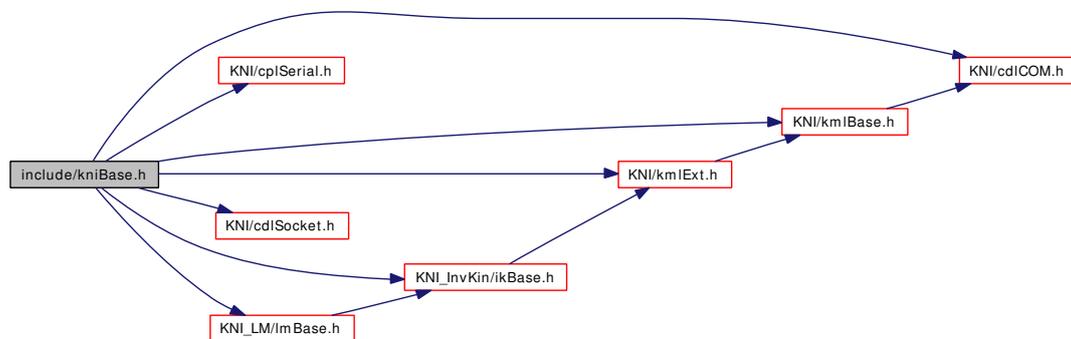
Classes

- struct [TLM_points](#)
[LM] linear movement: points to be interpolated
- struct [TLMtrajectory](#)
[LM] linear movement: parameters
- struct [TMLMIP](#)
[LM] Store intermediate targets for multiple linear movements
- struct [TPoint6D](#)
[LMBLEND] Standard coordinates for a point in space
- struct [TPoint3D](#)
- struct [TBlendtrace](#)
- struct [TSplinepoint](#)
- struct [TBLENDtrajectory](#)
[LMBLEND] Trajectory points
- class [JointSpeedException](#)
Joint speed too high.
- class [WaitParameterException](#)
Wait parameter set to false.
- class [CLMBase](#)
Linear movement Class.

12.26 include/kniBase.h File Reference

```
#include "KNI/cdlCOM.h"  
#include "KNI/cplSerial.h"  
#include "KNI/kmlBase.h"  
#include "KNI/kmlExt.h"  
#include "KNI/cdlSocket.h"  
#include "KNI_InvKin/ikBase.h"  
#include "KNI_LM/lmBase.h"
```

Include dependency graph for kniBase.h:



Index

- ~CCdlBase
 - CCdlBase, 37
- ~CCdlCOM
 - CCdlCOM, 41
- ~CCdlSocket
 - CCdlSocket, 45
- ~CCplBase
 - CCplBase, 48
- ~CKatBase
 - CKatBase, 79
- ~CKatana
 - CKatana, 69
- ~CMotBase
 - CMotBase, 98
- ~CSctBase
 - CSctBase, 121
- ~CikBase
 - CikBase, 60
- ~Exception
 - Exception, 130
- ~KatanaKinematics
 - KNI::KatanaKinematics, 134
- _ElapsedTime
 - KNI::Timer, 200
- _activatePositionController
 - CLMBase, 93
- _calibrationParameters
 - CMotBase, 105
- _ccd
 - CCdlCOM, 41
- _configfile
 - KNI::kmlFactory, 169
- _ct
 - KNI::Timer, 200
- _deviceName
 - CCdlCOM, 41
- _encoderLimits
 - CMotBase, 105
- _error_number
 - Exception, 130
- _gripperCloseEncoders
 - CKatana, 75
- _gripperIsPresent
 - CKatana, 75
- _gripperOpenEncoders
 - CKatana, 75
- _initKinematics
 - CikBase, 61
- _initialParameters
 - CMotBase, 105
- _ipAddr
 - CCdlSocket, 45
- _isInitialized
 - CLMBase, 93
- _kinematicsImpl
 - CikBase, 62
- _kinematicsIsInitialized
 - CikBase, 62
- _len
 - CCdlSocket, 45
- _length
 - KNI::KatanaKinematics5M180, 139
 - KNI::KatanaKinematics6M180, 146
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 160
- _maximumVelocity
 - CLMBase, 93
- _message
 - Exception, 130
- _nrOfPossibleSolutions
 - KNI::KatanaKinematics5M180, 139
 - KNI::KatanaKinematics6M180, 146
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 161
- _oto
 - CCdlCOM, 41
- _parameters
 - KNI::KatanaKinematics5M180, 139
 - KNI::KatanaKinematics6M180, 146
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 160
- _port
 - CCdlSocket, 45
- _prtHdl
 - CCdlCOM, 41
- _readEntry
 - KNI::kmlFactory, 168
- _setLength
 - KNI::KatanaKinematics5M180, 138
 - KNI::KatanaKinematics6M180, 145

- KNI::KatanaKinematics6M90G, 152
- KNI::KatanaKinematics6M90T, 160
- _setParameters
 - KNI::KatanaKinematics5M180, 138
 - KNI::KatanaKinematics6M180, 145
 - KNI::KatanaKinematics6M90G, 152
 - KNI::KatanaKinematics6M90T, 160
- _socketAddr
 - CCdlSocket, 46
- _socketfd
 - CCdlSocket, 46
- _timeout
 - KNI::Timer, 200
- _tolerance
 - KNI::KatanaKinematics5M180, 139
 - KNI::KatanaKinematics6M180, 146
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 160
- acotan
 - KNI_MHF, 29
- actcurr
 - TMotDYL, 222
- actpos
 - TMotAPS, 217
- adr
 - TKatGNL, 205
- AI
 - TPoint6D, 241
- angledef
 - KNI::KatanaKinematics6M180, 146
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 160
- angleOffset
 - KNI::KinematicParameters, 165
 - TMotInit, 228
- AnglePositionTest
 - KNI::KatanaKinematics6M180, 146
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 160
- angleRange
 - TMotInit, 228
- anglereduce
 - KNI_MHF, 29
- angles
 - KNI::KatanaKinematics, 134
- angles_container
 - KNI::KatanaKinematics5M180, 137
 - KNI::KatanaKinematics6M180, 145
 - KNI::KatanaKinematics6M90G, 152
 - KNI::KatanaKinematics6M90T, 159
- angleStop
 - KNI::KinematicParameters, 165
 - TMotInit, 228
- aps
 - CMotBase, 104
- arr
 - TKatMOT, 209
 - TKatSCT, 211
 - TSctDAT, 243
- arr_actpos
 - TLMtrajectory, 214
- arr_segment
 - TKatEFF, 204
- arr_tarpos
 - TLMtrajectory, 214
- atan0
 - KNI_MHF, 30
- atan1
 - KNI_MHF, 30
- b1
 - KNI::KatanaKinematics5M180::angles_calc, 140
 - KNI::KatanaKinematics6M180::angles_calc, 147
 - KNI::KatanaKinematics6M90G::angles_calc, 154
 - KNI::KatanaKinematics6M90T::angles_calc, 163
 - TBlendtrace, 192
- b2
 - KNI::KatanaKinematics5M180::angles_calc, 141
 - KNI::KatanaKinematics6M180::angles_calc, 148
 - KNI::KatanaKinematics6M90G::angles_calc, 155
 - KNI::KatanaKinematics6M90T::angles_calc, 163
 - TBlendtrace, 192
- base
 - CKatana, 75
- baud
 - TCdlCOMDesc, 196
- Be
 - TPoint6D, 241
- blendtrace
 - TBLENDtrajectory, 195
- blendtrajectory
 - CLMBase, 93
- blocked
 - CMotBase, 106
- byte
 - cdlBase.h, 267
 - cplBase.h, 272
 - kmlBase.h, 279
 - kmlCommon.h, 281

- BYTE_DECLARED
 - cdlBase.h, 267
 - cplBase.h, 272
 - kmlBase.h, 278
 - kmlCommon.h, 281
- c_iter
 - KNI::KinematicsDefaultEncMinAlgorithm, 166
- calcParameters
 - CLMBase, 89
- calibrate
 - CKatana, 70
- CannotGetSetPortAttributesException, 33
 - CannotGetSetPortAttributesException, 34
- CannotGetSetPortAttributesException
 - CannotGetSetPortAttributesException, 34
- CannotOpenPortException, 35
 - CannotOpenPortException, 36
- CannotOpenPortException
 - CannotOpenPortException, 36
- cbx
 - CKatBase, 84
- CCdlBase, 37
- CCdlBase
 - ~CCdlBase, 37
 - recv, 38
 - send, 38
- CCdlCOM, 39
 - CCdlCOM, 41
- CCdlCOM
 - ~CCdlCOM, 41
 - _ccd, 41
 - _deviceName, 41
 - _oto, 41
 - _prtHdl, 41
 - CCdlCOM, 41
 - digit, 41
 - recv, 41
 - send, 41
- CCdlSocket, 43
 - CCdlSocket, 44
- CCdlSocket
 - ~CCdlSocket, 45
 - _ipAddr, 45
 - _len, 45
 - _port, 45
 - _socketAddr, 46
 - _socketfd, 46
 - CCdlSocket, 44
 - digit, 45
 - disconnect, 45
 - recv, 45
 - send, 45
- CCplBase, 47
- CCplBase
 - ~CCplBase, 48
 - comm, 48
 - device, 49
 - getMasterFirmware, 49
 - init, 48
 - mMasterRevision, 49
 - mMasterVersion, 49
- CCplSerial, 50
- CCplSerial
 - cmd, 51
 - defineProtocol, 51
 - hdr, 51
 - load_tbl, 51
 - read_buf, 52
 - send_buf, 52
- CCplSerialCRC, 53
- CCplSerialCRC
 - comm, 55
 - defineProtocol, 55
 - getMasterFirmware, 55
 - init, 55
 - load_tbl, 55
 - recv, 55
 - send, 55
- cdlBase.h
 - byte, 267
 - BYTE_DECLARED, 267
- cdlCOMExceptions.h
 - ERR_AXIS_ANY, 270
 - ERR_AXIS_COLLISION, 270
 - ERR_AXIS_HEARTBEAT, 270
 - ERR_AXIS_MOVE, 270
 - ERR_AXIS_MOVE_POLY, 270
 - ERR_AXIS_OPERATIONAL, 270
 - ERR_CRC, 270
 - ERR_FAILED, 270
 - ERR_INVALID_ARGUMENT, 270
 - ERR_MESSAGE, 270
 - ERR_MESSAGE_STRING, 270
 - ERR_PERIPHERAL, 270
 - ERR_RANGE_MISMATCH, 270
 - ERR_STATE_MISMATCH, 270
 - ERR_TYPE_MISMATCH, 270
- checkAngleInRange
 - CMotBase, 102
- checkEncoderInRange
 - CMotBase, 102
- checkENLD
 - CKatana, 71
- checkJointSpeed
 - CLMBase, 90
- checkKatanaType

- CKatBase, 82
- CikBase, 57
 - CikBase, 60
- CikBase
 - ~CikBase, 60
 - _initKinematics, 61
 - _kinematicsImpl, 62
 - _kinematicsIsInitialized, 62
 - CikBase, 60
 - DKApos, 61
 - getCoordinates, 61
 - IKCalculate, 61
 - IKGoto, 61
 - moveRobotTo, 61, 62
- CKatana, 63
 - ~CKatana, 69
 - _gripperCloseEncoders, 75
 - _gripperIsPresent, 75
 - _gripperOpenEncoders, 75
 - base, 75
 - calibrate, 70
 - checkENLD, 71
 - CKatana, 69
 - closeGripper, 74
 - create, 69, 70
 - dec, 70
 - decDegrees, 71
 - disableCrashLimits, 72
 - enableCrashLimits, 72
 - freezeMotor, 74
 - freezeRobot, 74
 - GetBase, 69
 - getMotorAccelerationLimit, 73
 - getMotorEncoders, 72
 - getMotorVelocityLimit, 73
 - getNumberOfMotors, 72
 - getRobotEncoders, 72
 - inc, 70
 - incDegrees, 71
 - mKatanaType, 75
 - mov, 70
 - movDegrees, 71
 - moveMotorBy, 73
 - moveMotorByEnc, 73
 - moveMotorTo, 73
 - moveMotorToEnc, 73
 - moveRobotToEnc, 73
 - moveRobotToEnc4D, 74
 - openGripper, 74
 - resetTPSP, 71
 - searchMechStop, 70
 - sendFourSplinesToMotor, 74, 75
 - sendSplineToMotor, 74
 - sendTPSP, 71
 - setCrashLimit, 72
 - setGripperParameters, 71
 - setMotorAccelerationLimit, 73
 - setMotorVelocityLimit, 73
 - setPositionCollisionLimit, 72
 - setRobotAccelerationLimit, 73
 - setRobotVelocityLimit, 73
 - setSpeedCollisionLimit, 72
 - setTolerance, 69
 - setTPSP, 71
 - setTPSPDegrees, 71
 - startFourSplinesMovement, 74
 - startSplineMovement, 74
 - switchMotorOff, 74
 - switchMotorOn, 74
 - switchRobotOff, 74
 - switchRobotOn, 74
 - unBlock, 72
 - waitForMotor, 73
- CKatBase, 76
 - CKatBase, 79
 - CMotBase, 104
 - CSctBase, 121
- CKatBase
 - ~CKatBase, 79
 - cbx, 84
 - checkKatanaType, 82
 - CKatBase, 79
 - ctb, 84
 - disableCrashLimits, 82
 - ech, 84
 - eff, 84
 - enableCrashLimits, 82
 - GetCBX, 80
 - GetCTB, 80
 - GetECH, 80
 - GetEFF, 81
 - GetGNL, 80
 - GetIDS, 80
 - getMasterFirmware, 82
 - GetMFW, 80
 - GetMOT, 80
 - getProtocol, 82
 - GetSCT, 80
 - gnl, 83
 - ids, 84
 - init, 81
 - mfw, 83
 - mMasterRevision, 85
 - mMasterVersion, 84
 - mot, 84
 - protocol, 84
 - recvCBX, 81
 - recvCTB, 81

- recvECH, 81
- recvGMS, 81
- recvIDS, 81
- recvMFW, 81
- recvMPS, 82
- recvNMP, 81
- sct, 84
- sendCBX, 82
- sendSLM, 83
- sendSLMP, 83
- sendTPSP, 82
- setCrashLimit, 82
- setPositionCollisionLimit, 83
- setSpeedCollisionLimit, 83
- startFourSplinesMovement, 83
- startSplineMovement, 83
- unBlock, 82
- CLMBase, 86
 - _activatePositionController, 93
 - _isInitialized, 93
 - _maximumVelocity, 93
 - blendtrajectory, 93
 - calcParameters, 89
 - checkJointSpeed, 90
 - CLMBase, 89
 - fillPoints, 89
 - getActivatePositionController, 92
 - getMaximumLinearVelocity, 92
 - initLM, 90
 - moveRobotLinearTo, 92
 - movLM, 91
 - movLM2P, 91
 - movLM2P4D, 91
 - movLM2PwithL, 91
 - polCoefficients, 89
 - polDeviratives, 89
 - relPosition, 89
 - setActivatePositionController, 92
 - setMaximumLinearVelocity, 92
 - splineCoefficients, 90
 - totalTime, 89
 - trajectory, 93
- closeGripper
 - CKatana, 74
- cmd
 - CCplSerial, 51
- cmdtbl
 - TKatCTB, 202
- CMotBase, 94
- CMotBase
 - ~CMotBase, 98
 - _calibrationParameters, 105
 - _encoderLimits, 105
 - _initialParameters, 105
 - aps, 104
 - blocked, 106
 - checkAngleInRange, 102
 - checkEncoderInRange, 102
 - CKatBase, 104
 - dec, 102
 - decDegrees, 102
 - dyl, 105
 - freedom, 105
 - GetAPS, 99
 - GetBlocked, 100
 - GetCLB, 99
 - GetDYL, 99
 - GetEncoderMaxPos, 100
 - GetEncoderMinPos, 99
 - GetEncoderRange, 100
 - GetEncoderTolerance, 99
 - GetFreedom, 100
 - GetGNL, 99
 - GetInitialParameters, 99
 - GetNmp, 100
 - getParameterOrLimit, 104
 - GetPVP, 99
 - GetSCP, 99
 - GetSFW, 99
 - GetTPS, 99
 - gnl, 104
 - inc, 102
 - incDegrees, 102
 - init, 100
 - mov, 102
 - movDegrees, 102
 - nmp, 105
 - protocol, 106
 - pvp, 105
 - recvDYL, 101
 - recvPVP, 101
 - recvSCP, 101
 - recvSFW, 101
 - resetBlocked, 103
 - resetTPSP, 101
 - scp, 105
 - sendAPS, 100
 - sendDYL, 100
 - sendFourSplines, 103
 - sendSCP, 100
 - sendSpline, 103
 - sendTPS, 100
 - setAccelerationLimit, 103
 - setCalibrated, 102
 - setCalibrationParameters, 102
 - setControllerParameters, 103
 - setCrashLimit, 103
 - setCrashLimitLinear, 104

- setDYL, 101
- setInitialParameters, 101
- setPositionCollisionLimit, 104
- setPwmLimits, 103
- setSCP, 101
- setSpeedCollisionLimit, 104
- setSpeedLimit, 103
- setSpeedLimits, 103
- setTolerance, 102
- setTPSP, 101
- setTPSPDegrees, 101
- sfw, 105
- tps, 104
- waitForMotor, 102
- cnt
 - TKatMOT, 209
 - TKatSCT, 210
 - TSctDAT, 243
- coefficients
 - TLMtrajectory, 215
- comm
 - CCplBase, 48
 - CCplSerialCRC, 55
- ConfigFileEntryNotFoundException, 107
 - ConfigFileEntryNotFoundException, 108
- ConfigFileEntryNotFoundException
 - ConfigFileEntryNotFoundException, 108
- ConfigFileOpenException, 109
 - ConfigFileOpenException, 110
- ConfigFileOpenException
 - ConfigFileOpenException, 110
- ConfigFileSectionNotFoundException, 111
 - ConfigFileSectionNotFoundException, 112
- ConfigFileSectionNotFoundException
 - ConfigFileSectionNotFoundException, 112
- ConfigFileStateException, 113
 - ConfigFileStateException, 114
- ConfigFileStateException
 - ConfigFileStateException, 114
- ConfigFileSubsectionNotFoundException, 115
 - ConfigFileSubsectionNotFoundException, 116
- ConfigFileSubsectionNotFoundException
 - ConfigFileSubsectionNotFoundException, 116
- ConfigFileSyntaxErrorException, 117
 - ConfigFileSyntaxErrorException, 118
- ConfigFileSyntaxErrorException
 - ConfigFileSyntaxErrorException, 118
- Context, 119
 - Context, 119
- coordinates
 - KNI::KatanaKinematics, 134
- costh3
 - KNI::KatanaKinematics5M180::angles_calc, 141
 - KNI::KatanaKinematics6M180::angles_calc, 148
 - KNI::KatanaKinematics6M90G::angles_calc, 155
 - KNI::KatanaKinematics6M90T::angles_calc, 163
- cplBase.h
 - byte, 272
 - BYTE_DECLARED, 272
- cplSerial.h
 - KATANA_ERROR_FLAG, 275
 - NUMBER_OF_RETRIES_RECV, 275
 - NUMBER_OF_RETRIES_SEND, 275
- crash_limit_lin_nmp
 - TMotSCP, 234
- crash_limit_nmp
 - TMotSCP, 234
- CRC.h
 - CRC_CHECKSUM, 276
 - uint16, 276
 - uint8, 276
- CRC_CHECKSUM
 - CRC.h, 276
- create
 - CKatana, 69, 70
- CSctBase, 120
- CSctBase
 - ~CSctBase, 121
 - CKatBase, 121
 - dat, 121
 - GetDAT, 121
 - GetGNL, 121
 - gnl, 121
 - init, 121
 - protocol, 122
 - recvDAT, 121
- ctb
 - CKatBase, 84
- ctrlID
 - TSctDesc, 244
- dat
 - CSctBase, 121
- data
 - TCdlCOMDesc, 196
 - THeader, 198
- dec
 - CKatana, 70
 - CMotBase, 102
- decDegrees
 - CKatana, 71
 - CMotBase, 102
- defineProtocol
 - CCplSerial, 51

- CCplSerialCRC, 55
- deg2rad
 - KNI_MHF, 30
- derivatives
 - TLMtrajectory, 215
- desc
 - TKatMOT, 209
 - TKatSCT, 211
- device
 - CCplBase, 49
- DeviceReadException, 123
 - DeviceReadException, 124
- DeviceReadException
 - DeviceReadException, 124
- DeviceWriteException, 125
 - DeviceWriteException, 126
- DeviceWriteException
 - DeviceWriteException, 126
- digit
 - CCdlCOM, 41
 - CCdlSocket, 45
- dir
 - TMotCLB, 219
- DIR_NEGATIVE
 - kmlMotBase.h, 286
- DIR_POSITIVE
 - kmlMotBase.h, 286
- disableCrashLimits
 - CKatana, 72
 - CKatBase, 82
- disconnect
 - CCdlSocket, 45
- distance
 - TLMtrajectory, 214
- distBA
 - TBlendtrace, 193
- DK
 - KNI::KatanaKinematics, 135
 - KNI::KatanaKinematics5M180, 138
 - KNI::KatanaKinematics6M180, 145
 - KNI::KatanaKinematics6M90G, 152
 - KNI::KatanaKinematics6M90T, 159
- DKApos
 - CikBase, 61
- DLLDIR
 - dllexport.h, 262
- DLLDIR_IK
 - dllexport.h, 262
- DLLDIR_LM
 - dllexport.h, 262
- dllexport.h
 - DLLDIR, 262
 - DLLDIR_IK, 262
 - DLLDIR_LM, 262
- dt
 - TLMtrajectory, 214
- dyl
 - CMotBase, 105
 - TMotCLB, 219
- ech
 - CKatBase, 84
- echo
 - TKatECH, 203
- eff
 - CKatBase, 84
- Elapsed
 - KNI::Timer, 200
- ElapsedTime
 - KNI::Timer, 200
- enable
 - TMotCLB, 219
- enableCrashLimits
 - CKatana, 72
 - CKatBase, 82
- enc2rad
 - KNI_MHF, 30
- enc_maxpos
 - TMotENL, 224
- enc_minpos
 - TMotENL, 224
- enc_per_cycle
 - TMotENL, 224
- enc_range
 - TMotENL, 224
- enc_tolerance
 - TMotENL, 225
- encoderOffset
 - TMotInit, 228
- encoderPositionAfter
 - TMotCLB, 219
- encoders
 - KNI::KatanaKinematics, 134
 - KNI::KinematicsDefaultEncMinAlgorithm, 166
- encodersPerCycle
 - TMotInit, 228
- encOffset
 - KNI::KinematicParameters, 165
- epc
 - KNI::KinematicParameters, 165
- ERR_AXIS_ANY
 - cdlCOMExceptions.h, 270
- ERR_AXIS_COLLISION
 - cdlCOMExceptions.h, 270
- ERR_AXIS_HEARTBEAT
 - cdlCOMExceptions.h, 270
- ERR_AXIS_MOVE

- cdlCOMExceptions.h, 270
- ERR_AXIS_MOVE_POLY
 - cdlCOMExceptions.h, 270
- ERR_AXIS_OPERATIONAL
 - cdlCOMExceptions.h, 270
- ERR_CRC
 - cdlCOMExceptions.h, 270
- ERR_FAILED
 - cdlCOMExceptions.h, 270
- ERR_INVALID_ARGUMENT
 - cdlCOMExceptions.h, 270
- ERR_MESSAGE
 - cdlCOMExceptions.h, 270
- ERR_MESSAGE_STRING
 - cdlCOMExceptions.h, 270
- ERR_PERIPHERAL
 - cdlCOMExceptions.h, 270
- ERR_RANGE_MISMATCH
 - cdlCOMExceptions.h, 270
- ERR_STATE_MISMATCH
 - cdlCOMExceptions.h, 270
- ERR_TYPE_MISMATCH
 - cdlCOMExceptions.h, 270
- error_number
 - Exception, 130
- ErrorException, 127
 - ErrorException, 128
- ErrorException
 - ErrorException, 128
- Exception, 129
 - ~Exception, 130
 - _error_number, 130
 - _message, 130
 - error_number, 130
 - Exception, 130
 - message, 130
 - what, 130
- Exceptions, 19
- fillPoints
 - CLMBase, 89
- findFirstEqualAngle
 - KNI::KatanaKinematics6M90T, 160
 - KNI_MHF, 30
- freedom
 - CMotBase, 105
- freezeMotor
 - CKatana, 74
- freezeRobot
 - CKatana, 74
- Ga
 - TPoint6D, 241
- getActivatePositionController
 - CLMBase, 92
- GetAPS
 - CMotBase, 99
- GetBase
 - CKatana, 69
- GetBlocked
 - CMotBase, 100
- GetCBX
 - CKatBase, 80
- GetCLB
 - CMotBase, 99
- getCoordinates
 - CikBase, 61
- GetCTB
 - CKatBase, 80
- GetDAT
 - CSctBase, 121
- GetDYL
 - CMotBase, 99
- GetECH
 - CKatBase, 80
- GetEFF
 - CKatBase, 81
- getEFF
 - KNI::kmlFactory, 169
- GetEncoderMaxPos
 - CMotBase, 100
- GetEncoderMinPos
 - CMotBase, 99
- GetEncoderRange
 - CMotBase, 100
- GetEncoderTolerance
 - CMotBase, 99
- GetFreedom
 - CMotBase, 100
- GetGNL
 - CKatBase, 80
 - CMotBase, 99
 - CSctBase, 121
- getGNL
 - KNI::kmlFactory, 168
- getGripperParameters
 - KNI::kmlFactory, 169
- GetIDS
 - CKatBase, 80
- GetInitialParameters
 - CMotBase, 99
- getMasterFirmware
 - CCplBase, 49
 - CCplSerialCRC, 55
 - CKatBase, 82
- getMaximumLinearVelocity
 - CLMBase, 92
- GetMFW

- CKatBase, 80
- GetMOT
 - CKatBase, 80
- getMOT
 - KNI::kmlFactory, 169
- getMotCLB
 - KNI::kmlFactory, 169
- getMotDesc
 - KNI::kmlFactory, 169
- getMotDYL
 - KNI::kmlFactory, 169
- getMotInit
 - KNI::kmlFactory, 169
- getMotorAccelerationLimit
 - CKatana, 73
- getMotorEncoders
 - CKatana, 72
- getMotorVelocityLimit
 - CKatana, 73
- getMotSCP
 - KNI::kmlFactory, 169
- GetNmp
 - CMotBase, 100
- getNumberOfMotors
 - CKatana, 72
- getParameterOrLimit
 - CMotBase, 104
- getProtocol
 - CKatBase, 82
- GetPVP
 - CMotBase, 99
- getRobotEncoders
 - CKatana, 72
- GetSCP
 - CMotBase, 99
- GetSCT
 - CKatBase, 80
- getSCT
 - KNI::kmlFactory, 169
- getSctDesc
 - KNI::kmlFactory, 169
- GetSFW
 - CMotBase, 99
- GetTPS
 - CMotBase, 99
- getType
 - KNI::kmlFactory, 169
- gnl
 - CKatBase, 83
 - CMotBase, 104
 - CSctBase, 121
- GripperTest
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 160
- hdr
 - CCplSerial, 51
- ids
 - CKatBase, 84
- IK
 - KNI::KatanaKinematics, 135
 - KNI::KatanaKinematics5M180, 138
 - KNI::KatanaKinematics6M180, 145
 - KNI::KatanaKinematics6M90G, 152
 - KNI::KatanaKinematics6M90T, 159
- IK_b1b2cosh3_6M180
 - KNI::KatanaKinematics6M180, 146
- IK_b1b2cosh3_6MS
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 160
- IK_theta234theta5
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 160
- ikBase.h
 - TM_ENDLESS, 288
- IKCalculate
 - CikBase, 61
- IKGoto
 - CikBase, 61
- inc
 - CKatana, 70
 - CMotBase, 102
- incDegrees
 - CKatana, 71
 - CMotBase, 102
- include/ Directory Reference, 22
- include/common/ Directory Reference, 21
- include/common/dllexport.h, 261
- include/common/exception.h, 263
- include/common/MathHelperFunctions.h, 264
- include/common/Timer.h, 266
- include/KNI/ Directory Reference, 23
- include/KNI/cdlBase.h, 267
- include/KNI/cdlCOM.h, 268
- include/KNI/cdlCOMExceptions.h, 269
- include/KNI/cdlSocket.h, 271
- include/KNI/cplBase.h, 272
- include/KNI/cplSerial.h, 274
- include/KNI/CRC.h, 276
- include/KNI/kmlBase.h, 277
- include/KNI/kmlCommon.h, 280
- include/KNI/kmlExt.h, 282
- include/KNI/kmlFactories.h, 283
- include/KNI/kmlMotBase.h, 284
- include/KNI/kmlSctBase.h, 287
- include/KNI_InvKin/ Directory Reference, 24
- include/KNI_InvKin/ikBase.h, 288
- include/KNI_InvKin/KatanaKinematics.h, 289

- include/KNI_InvKin/KatanaKinematics5M180.h, 290
- include/KNI_InvKin/KatanaKinematics6M180.h, 291
- include/KNI_InvKin/KatanaKinematics6M90G.h, 292
- include/KNI_InvKin/KatanaKinematics6M90T.h, 293
- include/KNI_InvKin/KatanaKinematicsDecisionAlgorithmFactory, 294
- include/KNI_LM/ Directory Reference, 25
- include/KNI_LM/lmBase.h, 295
- include/kniBase.h, 296
- init
 - CCplBase, 48
 - CCplSerialCRC, 55
 - CKatBase, 81
 - CMotBase, 100
 - CSctBase, 121
 - KNI::KatanaKinematics, 135
 - KNI::KatanaKinematics5M180, 138
 - KNI::KatanaKinematics6M180, 145
 - KNI::KatanaKinematics6M90G, 152
 - KNI::KatanaKinematics6M90T, 159
- initLM
 - CLMBase, 90
- inp
 - TKatCBX, 201
- isCalibrated
 - TMotCLB, 219
- JointSpeedException, 131
 - JointSpeedException, 132
- JointSpeedException
 - JointSpeedException, 132
- K400_OLD_PROTOCOL_THRESHOLD
 - kmlBase.h, 278
- kARW
 - TMotSCP, 233
- KATANA_ERROR_FLAG
 - cplSerial.h, 275
- kD
 - TMotSCP, 233
- kD_speed
 - TMotSCP, 234
- kI
 - TMotSCP, 233
- kI_nmp
 - TMotSCP, 234
- kI_speed
 - TMotSCP, 234
- kmlBase.h
 - byte, 279
 - BYTE_DECLARED, 278
 - K400_OLD_PROTOCOL_THRESHOLD, 278
 - TM_ENDLESS, 278
- kmlCommon.h
 - byte, 281
 - BYTE_DECLARED, 281
 - TM_ENDLESS, 281
- kmlFactory
 - KNI::kmlFactory, 168
- kmlMotBase.h
 - DIR_NEGATIVE, 286
 - DIR_POSITIVE, 286
 - MCF_CALIB, 285
 - MCF_FREEZE, 285
 - MCF_OFF, 285
 - MCF_ON, 285
 - MSF_DESPOS, 286
 - MSF_LINMOV, 286
 - MSF_MAXPOS, 286
 - MSF_MECHSTOP, 286
 - MSF_MINPOS, 286
 - MSF_MOTCRASHED, 286
 - MSF_NLINMOV, 286
 - MSF_NORMOPSTAT, 286
 - MSF_NOTVALID, 286
- kmlMotBase.h
 - TMotCmdFlg, 285
 - TMotStsFlg, 285
 - TSearchDir, 286
- KNI, 27
 - sleep, 28
- KNI::KatanaKinematics, 133
- KNI::KatanaKinematics
 - ~KatanaKinematics, 134
 - angles, 134
 - coordinates, 134
 - DK, 135
 - encoders, 134
 - IK, 135
 - init, 135
 - metrics, 134
 - parameter_container, 134
- KNI::KatanaKinematics5M180, 136
- KNI::KatanaKinematics5M180
 - _length, 139
 - _nrOfPossibleSolutions, 139
 - _parameters, 139
 - _setLength, 138
 - _setParameteres, 138
 - _tolerance, 139
 - angles_container, 137
 - DK, 138
 - IK, 138

- init, 138
- KNI::KatanaKinematics5M180::angles_calc, 140
- KNI::KatanaKinematics5M180::angles_calc
 - b1, 140
 - b2, 141
 - costh3, 141
 - theta1, 140
 - theta2, 140
 - theta234, 140
 - theta3, 140
 - theta4, 140
 - theta5, 140
- KNI::KatanaKinematics5M180::position, 142
- KNI::KatanaKinematics5M180::position
 - x, 142
 - y, 142
 - z, 142
- KNI::KatanaKinematics6M180, 143
- KNI::KatanaKinematics6M180
 - _length, 146
 - _nrOfPossibleSolutions, 146
 - _parameters, 146
 - _setLength, 145
 - _setParameters, 145
 - _tolerance, 146
 - angledef, 146
 - AnglePositionTest, 146
 - angles_container, 145
 - DK, 145
 - IK, 145
 - IK_b1b2costh3_6M180, 146
 - init, 145
 - PositionTest6M180, 146
 - thetacomp, 146
- KNI::KatanaKinematics6M180::angles_calc, 147
- KNI::KatanaKinematics6M180::angles_calc
 - b1, 147
 - b2, 148
 - costh3, 148
 - theta1, 147
 - theta2, 147
 - theta234, 147
 - theta3, 147
 - theta4, 147
 - theta5, 147
- KNI::KatanaKinematics6M180::position, 149
- KNI::KatanaKinematics6M180::position
 - x, 149
 - y, 149
 - z, 149
- KNI::KatanaKinematics6M90G, 150
- KNI::KatanaKinematics6M90G
 - _length, 153
 - _nrOfPossibleSolutions, 153
 - _parameters, 153
 - _setLength, 152
 - _setParameters, 152
 - _tolerance, 153
 - angledef, 153
 - AnglePositionTest, 153
 - angles_container, 152
 - DK, 152
 - GripperTest, 153
 - IK, 152
 - IK_b1b2costh3_6MS, 153
 - IK_theta234theta5, 153
 - init, 152
 - PositionTest6MS, 153
 - thetacomp, 153
- KNI::KatanaKinematics6M90G::angles_calc, 154
- KNI::KatanaKinematics6M90G::angles_calc
 - b1, 154
 - b2, 155
 - costh3, 155
 - theta1, 154
 - theta2, 154
 - theta234, 154
 - theta3, 154
 - theta4, 154
 - theta5, 154
- KNI::KatanaKinematics6M90G::position, 156
- KNI::KatanaKinematics6M90G::position
 - x, 156
 - y, 156
 - z, 156
- KNI::KatanaKinematics6M90T, 157
- KNI::KatanaKinematics6M90T
 - _length, 160
 - _nrOfPossibleSolutions, 161
 - _parameters, 160
 - _setLength, 160
 - _setParameters, 160
 - _tolerance, 160
 - angledef, 160
 - AnglePositionTest, 160
 - angles_container, 159
 - DK, 159
 - findFirstEqualAngle, 160
 - GripperTest, 160
 - IK, 159
 - IK_b1b2costh3_6MS, 160
 - IK_theta234theta5, 160
 - init, 159
 - PositionTest6MS, 160
 - thetacomp, 160
- KNI::KatanaKinematics6M90T::angles_calc, 162
- KNI::KatanaKinematics6M90T::angles_calc
 - b1, 163

- b2, 163
- costh3, 163
- theta1, 162
- theta2, 162
- theta234, 162
- theta3, 162
- theta4, 162
- theta5, 162
- theta6, 162
- KNI::KatanaKinematics6M90T::position, 164
- KNI::KatanaKinematics6M90T::position
 - x, 164
 - y, 164
 - z, 164
- KNI::KinematicParameters, 165
- KNI::KinematicParameters
 - angleOffset, 165
 - angleStop, 165
 - encOffset, 165
 - epc, 165
 - rotDir, 165
- KNI::KinematicsDefaultEncMinAlgorithm, 166
- KNI::KinematicsDefaultEncMinAlgorithm
 - c_iter, 166
 - encoders, 166
 - operator(), 166
 - t_iter, 166
- KNI::kmlFactory, 167
- KNI::kmlFactory
 - _configfile, 169
 - _readEntry, 168
 - getEFF, 169
 - getGNL, 168
 - getGripperParameters, 169
 - getMOT, 169
 - getMotCLB, 169
 - getMotDesc, 169
 - getMotDYL, 169
 - getMotInit, 169
 - getMotSCP, 169
 - getSCT, 169
 - getSctDesc, 169
 - getType, 169
 - kmlFactory, 168
 - openFile, 168
- KNI::NoSolutionException, 176
- KNI::NoSolutionException
 - NoSolutionException, 177
- KNI::Timer, 199
 - _ElapsedTime, 200
 - _ct, 200
 - _timeout, 200
 - Elapsed, 200
 - ElapsedTime, 200
 - Set, 200
 - Set_And_Start, 200
 - Start, 200
 - Timer, 200
 - WaitUntilElapsed, 200
- KNI_MHF, 29
 - acotan, 29
 - anglereducer, 29
 - atan0, 30
 - atan1, 30
 - deg2rad, 30
 - enc2rad, 30
 - findFirstEqualAngle, 30
 - pow2, 31
 - rad2deg, 31
 - rad2enc, 31
 - sign, 31
- KNI_MHF::unary_deg2rad, 248
 - operator(), 248
- KNI_MHF::unary_precalc_cos, 249
 - operator(), 249
- KNI_MHF::unary_precalc_sin, 250
 - operator(), 250
- KNI_MHF::unary_rad2deg, 251
 - operator(), 251
- kP
 - TMotSCP, 233
- kP_speed
 - TMotSCP, 233
- kpos_nmp
 - TMotSCP, 234
- kspeed_nmp
 - TMotSCP, 234
- load_tbl
 - CCplSerial, 51
 - CCplSerialCRC, 55
- m1
 - TBlendtrace, 192
- m2
 - TBlendtrace, 192
- M_PI
 - MathHelperFunctions.h, 265
- MathHelperFunctions.h
 - M_PI, 265
- maxaccel
 - TMotDYL, 222
- maxaccel_nmp
 - TMotDYL, 222
- maxcurr
 - TMotDYL, 222
- maxcurr_nmp
 - TMotDYL, 223

- maxdecel
 - TMotDYL, [222](#)
- maxnpwm
 - TMotSCP, [233](#)
- maxnpwm_nmp
 - TMotSCP, [234](#)
- maxnspeed
 - TMotDYL, [222](#)
- maxnspeed_nmp
 - TMotDYL, [223](#)
- maxppwm
 - TMotSCP, [233](#)
- maxppwm_nmp
 - TMotSCP, [234](#)
- maxpspeed
 - TMotDYL, [222](#)
- maxpspeed_nmp
 - TMotDYL, [222](#)
- mcf
 - TMotCLB, [219](#)
- MCF_CALIB
 - kmlMotBase.h, [285](#)
- MCF_FREEZE
 - kmlMotBase.h, [285](#)
- MCF_OFF
 - kmlMotBase.h, [285](#)
- MCF_ON
 - kmlMotBase.h, [285](#)
- mcfAPS
 - TMotAPS, [217](#)
- mcfTPS
 - TMotTPS, [238](#)
- message
 - Exception, [130](#)
- metrics
 - KNI::KatanaKinematics, [134](#)
- mfw
 - CKatBase, [83](#)
- minpos
 - TMotDYL, [222](#)
- mKatanaType
 - CKatana, [75](#)
- mlm_intermediate_pos
 - TMLMIP, [216](#)
- mMasterRevision
 - CCplBase, [49](#)
 - CKatBase, [85](#)
- mMasterVersion
 - CCplBase, [49](#)
 - CKatBase, [84](#)
- modelName
 - TKatGNL, [205](#)
- mot
 - CKatBase, [84](#)
- MotorCrashException, [170](#)
 - MotorCrashException, [171](#)
- MotorCrashException
 - MotorCrashException, [171](#)
- MotorOutOfRangeException, [172](#)
 - MotorOutOfRangeException, [173](#)
- MotorOutOfRangeException
 - MotorOutOfRangeException, [173](#)
- motors
 - TLMtrajectory, [215](#)
- MotorTimeoutException, [174](#)
 - MotorTimeoutException, [175](#)
- MotorTimeoutException
 - MotorTimeoutException, [175](#)
- mov
 - CKatana, [70](#)
 - CMotBase, [102](#)
- movDegrees
 - CKatana, [71](#)
 - CMotBase, [102](#)
- moveMotorBy
 - CKatana, [73](#)
- moveMotorByEnc
 - CKatana, [73](#)
- moveMotorTo
 - CKatana, [73](#)
- moveMotorToEnc
 - CKatana, [73](#)
- moveRobotLinearTo
 - CLMBase, [92](#)
- moveRobotTo
 - CikBase, [61](#), [62](#)
- moveRobotToEnc
 - CKatana, [73](#)
- moveRobotToEnc4D
 - CKatana, [74](#)
- movLM
 - CLMBase, [91](#)
- movLM2P
 - CLMBase, [91](#)
- movLM2P4D
 - CLMBase, [91](#)
- movLM2PwithL
 - CLMBase, [91](#)
- msf
 - TMotPVP, [230](#)
- MSF_DESPOS
 - kmlMotBase.h, [286](#)
- MSF_LINMOV
 - kmlMotBase.h, [286](#)
- MSF_MAXPOS
 - kmlMotBase.h, [286](#)
- MSF_MECHSTOP
 - kmlMotBase.h, [286](#)

- MSF_MINPOS
 - kmlMotBase.h, 286
- MSF_MOTCRASHED
 - kmlMotBase.h, 286
- MSF_NLINMOV
 - kmlMotBase.h, 286
- MSF_NORMOPSTAT
 - kmlMotBase.h, 286
- MSF_NOTVALID
 - kmlMotBase.h, 286
- nmp
 - CMotBase, 105
- NoSolutionException
 - KNI::NoSolutionException, 177
- number_of_blends
 - TBLENDtrajectory, 195
- number_of_points
 - TLMtrajectory, 214
- number_of_referencepoints
 - TBLENDtrajectory, 195
- NUMBER_OF_RETRIES_RECV
 - cplSerial.h, 275
- NUMBER_OF_RETRIES_SEND
 - cplSerial.h, 275
- number_of_splinepoints
 - TBLENDtrajectory, 195
- number_of_splines
 - TBLENDtrajectory, 195
- openFile
 - KNI::kmlFactory, 168
- openGripper
 - CKatana, 74
- operator()
 - KNI::KinematicsDefaultEncMinAlgorithm, 166
 - KNI_MHF::unary_deg2rad, 248
 - KNI_MHF::unary_precalc_cos, 249
 - KNI_MHF::unary_precalc_sin, 250
 - KNI_MHF::unary_rad2deg, 251
- order
 - TMotCLB, 219
- out
 - TKatCBX, 201
- own
 - TMotGNL, 227
 - TSctGNL, 246
- P1A
 - TBlendtrace, 192
- P1B
 - TBlendtrace, 192
- p1p2n
 - TBlendtrace, 192
- p2p3n
 - TBlendtrace, 192
- parameter_container
 - KNI::KatanaKinematics, 134
- ParameterReadingException, 178
 - ParameterReadingException, 179
- ParameterReadingException
 - ParameterReadingException, 179
- parameters
 - TLMtrajectory, 215
- ParameterWritingException, 180
 - ParameterWritingException, 181
- ParameterWritingException
 - ParameterWritingException, 181
- parity
 - TCdlCOMDesc, 197
- point
 - TSplinepoint, 247
- points
 - TLMtrajectory, 215
- polCoefficients
 - CLMBase, 89
- polDeviratives
 - CLMBase, 89
- port
 - TCdlCOMDesc, 196
- PortNotOpenException, 182
 - PortNotOpenException, 183
- PortNotOpenException
 - PortNotOpenException, 183
- pos
 - TLM_points, 212
 - TMotPVP, 230
- PositionTest6M180
 - KNI::KatanaKinematics6M180, 146
- PositionTest6MS
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 160
- pow2
 - KNI_MHF, 31
- protocol
 - CKatBase, 84
 - CMotBase, 106
 - CSctBase, 122
- pvp
 - CMotBase, 105
- pwm
 - TMotPVP, 230
- rad2deg
 - KNI_MHF, 31
- rad2enc
 - KNI_MHF, 31

- read_buf
 - CCplSerial, 52
- read_sz
 - TPacket, 239
- ReadNotCompleteException, 184
 - ReadNotCompleteException, 185
- ReadNotCompleteException
 - ReadNotCompleteException, 185
- ReadWriteNotCompleteException, 187
 - ReadWriteNotCompleteException, 188
- ReadWriteNotCompleteException
 - ReadWriteNotCompleteException, 188
- recv
 - CCdlBase, 38
 - CCdlCOM, 41
 - CCdlSocket, 45
 - CCplSerialCRC, 55
- recvCBX
 - CKatBase, 81
- recvCTB
 - CKatBase, 81
- recvDAT
 - CSctBase, 121
- recvDYL
 - CMotBase, 101
- recvECH
 - CKatBase, 81
- recvGMS
 - CKatBase, 81
- recvIDS
 - CKatBase, 81
- recvMFW
 - CKatBase, 81
- recvMPS
 - CKatBase, 82
- recvNMP
 - CKatBase, 81
- recvPVP
 - CMotBase, 101
- recvSCP
 - CMotBase, 101
- recvSFW
 - CMotBase, 101
- referencepoints
 - TBLENDtrajectory, 195
- relPosition
 - CLMBase, 89
- res
 - TSctGNL, 246
- resetBlocked
 - CMotBase, 103
- resetTPSP
 - CKatana, 71
 - CMotBase, 101
- rev
 - TKatMFW, 207
- revision
 - TMotSFW, 236
- rotationDirection
 - TMotInit, 228
- rotDir
 - KNI::KinematicParameters, 165
- rttc
 - TCdlCOMDesc, 197
- scp
 - CMotBase, 105
 - TMotCLB, 219
- sct
 - CKatBase, 84
- searchMechStop
 - CKatana, 70
- send
 - CCdlBase, 38
 - CCdlCOM, 41
 - CCdlSocket, 45
 - CCplSerialCRC, 55
- send_buf
 - CCplSerial, 52
- send_sz
 - TPacket, 239
- sendAPS
 - CMotBase, 100
- sendCBX
 - CKatBase, 82
- sendDYL
 - CMotBase, 100
- sendFourSplines
 - CMotBase, 103
- sendFourSplinesToMotor
 - CKatana, 74, 75
- sendSCP
 - CMotBase, 100
- sendSLM
 - CKatBase, 83
- sendSLMP
 - CKatBase, 83
- sendSpline
 - CMotBase, 103
- sendSplineToMotor
 - CKatana, 74
- sendTPS
 - CMotBase, 100
- sendTPSP
 - CKatana, 71
 - CKatBase, 82
- sens_count
 - TSctDesc, 244

- sens_res
 - TSctDesc, [244](#)
- Set
 - KNI::Timer, [200](#)
- Set_And_Start
 - KNI::Timer, [200](#)
- setAccelerationLimit
 - CMotBase, [103](#)
- setActivatePositionController
 - CLMBase, [92](#)
- setCalibrated
 - CMotBase, [102](#)
- setCalibrationParameters
 - CMotBase, [102](#)
- setControllerParameters
 - CMotBase, [103](#)
- setCrashLimit
 - CKatana, [72](#)
 - CKatBase, [82](#)
 - CMotBase, [103](#)
- setCrashLimitLinear
 - CMotBase, [104](#)
- setDYL
 - CMotBase, [101](#)
- setGripperParameters
 - CKatana, [71](#)
- setInitialParameters
 - CMotBase, [101](#)
- setMaximumLinearVelocity
 - CLMBase, [92](#)
- setMotorAccelerationLimit
 - CKatana, [73](#)
- setMotorVelocityLimit
 - CKatana, [73](#)
- setPositionCollisionLimit
 - CKatana, [72](#)
 - CKatBase, [83](#)
 - CMotBase, [104](#)
- setPwmLimits
 - CMotBase, [103](#)
- setRobotAccelerationLimit
 - CKatana, [73](#)
- setRobotVelocityLimit
 - CKatana, [73](#)
- setSCP
 - CMotBase, [101](#)
- setSpeedCollisionLimit
 - CKatana, [72](#)
 - CKatBase, [83](#)
 - CMotBase, [104](#)
- setSpeedLimit
 - CMotBase, [103](#)
- setSpeedLimits
 - CMotBase, [103](#)
- setTolerance
 - CKatana, [69](#)
 - CMotBase, [102](#)
- setTPSP
 - CKatana, [71](#)
 - CMotBase, [101](#)
- setTPSPDegrees
 - CKatana, [71](#)
 - CMotBase, [101](#)
- sfw
 - CMotBase, [105](#)
- SID
 - TMotGNL, [227](#)
 - TSctGNL, [246](#)
- sign
 - KNI_MHF, [31](#)
- size
 - THeader, [198](#)
- SlaveErrorException, [189](#)
 - SlaveErrorException, [190](#)
- SlaveErrorException
 - SlaveErrorException, [190](#)
- sleep
 - KNI, [28](#)
- slvID
 - TMotDesc, [220](#)
- splineCoefficients
 - CLMBase, [90](#)
- splinepoints
 - TBLENDtrajectory, [195](#)
- Start
 - KNI::Timer, [200](#)
- startFourSplinesMovement
 - CKatana, [74](#)
 - CKatBase, [83](#)
- startSplineMovement
 - CKatana, [74](#)
 - CKatBase, [83](#)
- stop
 - TCdlCOMDesc, [197](#)
- strID
 - TKatIDS, [206](#)
- subtype
 - TMotSFW, [237](#)
- subversion
 - TMotSFW, [236](#)
- switchMotorOff
 - CKatana, [74](#)
- switchMotorOn
 - CKatana, [74](#)
- switchRobotOff
 - CKatana, [74](#)
- switchRobotOn
 - CKatana, [74](#)

- t_iter
 - KNI::KinematicsDefaultEncMinAlgorithm, 166
- tA
 - TBlendtrace, 192
- tarpos
 - TMotTPS, 238
- tB
 - TBlendtrace, 192
- TBlendtrace, 191
 - b1, 192
 - b2, 192
 - distBA, 193
 - m1, 192
 - m2, 192
 - P1A, 192
 - P1B, 192
 - p1p2n, 192
 - p2p3n, 192
 - tA, 192
 - tB, 192
 - V1A, 192
 - V1B, 192
- TBLENDtrajectory, 194
 - blendtrace, 195
 - number_of_blends, 195
 - number_of_referencepoints, 195
 - number_of_splinepoints, 195
 - number_of_splines, 195
 - referencepoints, 195
 - splinepoints, 195
 - tEnd, 195
- TCdlCOMDesc, 196
- TCdlCOMDesc
 - baud, 196
 - data, 196
 - parity, 197
 - port, 196
 - rttc, 197
 - stop, 197
 - wttc, 197
- tEnd
 - TBLENDtrajectory, 195
- THeader, 198
 - data, 198
 - size, 198
- theta1
 - KNI::KatanaKinematics5M180::angles_calc, 140
 - KNI::KatanaKinematics6M180::angles_calc, 147
 - KNI::KatanaKinematics6M90G::angles_calc, 154
- theta2
 - KNI::KatanaKinematics6M90T::angles_calc, 162
 - KNI::KatanaKinematics5M180::angles_calc, 140
 - KNI::KatanaKinematics6M180::angles_calc, 147
 - KNI::KatanaKinematics6M90G::angles_calc, 154
 - KNI::KatanaKinematics6M90T::angles_calc, 162
- theta234
 - KNI::KatanaKinematics5M180::angles_calc, 140
 - KNI::KatanaKinematics6M180::angles_calc, 147
 - KNI::KatanaKinematics6M90G::angles_calc, 154
 - KNI::KatanaKinematics6M90T::angles_calc, 162
- theta3
 - KNI::KatanaKinematics5M180::angles_calc, 140
 - KNI::KatanaKinematics6M180::angles_calc, 147
 - KNI::KatanaKinematics6M90G::angles_calc, 154
 - KNI::KatanaKinematics6M90T::angles_calc, 162
- theta4
 - KNI::KatanaKinematics5M180::angles_calc, 140
 - KNI::KatanaKinematics6M180::angles_calc, 147
 - KNI::KatanaKinematics6M90G::angles_calc, 154
 - KNI::KatanaKinematics6M90T::angles_calc, 162
- theta5
 - KNI::KatanaKinematics5M180::angles_calc, 140
 - KNI::KatanaKinematics6M180::angles_calc, 147
 - KNI::KatanaKinematics6M90G::angles_calc, 154
 - KNI::KatanaKinematics6M90T::angles_calc, 162
- theta6
 - KNI::KatanaKinematics6M90T::angles_calc, 162
- thetacomp
 - KNI::KatanaKinematics6M180, 146
 - KNI::KatanaKinematics6M90G, 153
 - KNI::KatanaKinematics6M90T, 160

- time
 - TLM_points, 212
 - TLMtrajectory, 214
 - TSplinepoint, 247
- Timer
 - KNI::Timer, 200
- TKatCBX, 201
- TKatCBX
 - inp, 201
 - out, 201
- TKatCTB, 202
- TKatCTB
 - cmdtbl, 202
- TKatECH, 203
- TKatECH
 - echo, 203
- TKatEFF, 204
- TKatEFF
 - arr_segment, 204
- TKatGNL, 205
- TKatGNL
 - adr, 205
 - modelName, 205
- TKatIDS, 206
- TKatIDS
 - strID, 206
- TKatMFW, 207
- TKatMFW
 - rev, 207
 - ver, 207
- TKatMOT, 208
- TKatMOT
 - arr, 209
 - cnt, 209
 - desc, 209
- TKatSCT, 210
- TKatSCT
 - arr, 211
 - cnt, 210
 - desc, 211
- TLM_points, 212
 - pos, 212
 - time, 212
- TLMtrajectory, 213
 - arr_actpos, 214
 - arr_tarpos, 214
 - coefficients, 215
 - derivatives, 215
 - distance, 214
 - dt, 214
 - motors, 215
 - number_of_points, 214
 - parameters, 215
 - points, 215
 - time, 214
- TM_ENDLESS
 - ikBase.h, 288
 - kmlBase.h, 278
 - kmlCommon.h, 281
- TMLMIP, 216
 - mlm_intermediate_pos, 216
- TMotAPS, 217
- TMotAPS
 - actpos, 217
 - mcfAPS, 217
- TMotCLB, 218
- TMotCLB
 - dir, 219
 - dyl, 219
 - enable, 219
 - encoderPositionAfter, 219
 - isCalibrated, 219
 - mcf, 219
 - order, 219
 - scp, 219
- TMotCmdFlg
 - kmlMotBase.h, 285
- TMotDesc, 220
- TMotDesc
 - slvID, 220
- TMotDYL, 221
- TMotDYL
 - actcurr, 222
 - maxaccel, 222
 - maxaccel_nmp, 222
 - maxcurr, 222
 - maxcurr_nmp, 223
 - maxdecel, 222
 - maxnspeed, 222
 - maxnspeed_nmp, 223
 - maxpspeed, 222
 - maxpspeed_nmp, 222
 - minpos, 222
- TMotENL, 224
- TMotENL
 - enc_maxpos, 224
 - enc_minpos, 224
 - enc_per_cycle, 224
 - enc_range, 224
 - enc_tolerance, 225
- TMotGNL, 226
- TMotGNL
 - own, 227
 - SID, 227
- TMotInit, 228
- TMotInit
 - angleOffset, 228
 - angleRange, 228

- angleStop, 228
- encoderOffset, 228
- encodersPerCycle, 228
- rotationDirection, 228
- TMotPVP, 230
- TMotPVP
 - msf, 230
 - pos, 230
 - pwm, 230
 - vel, 230
- TMotSCP, 232
- TMotSCP
 - crash_limit_lin_nmp, 234
 - crash_limit_nmp, 234
 - kARW, 233
 - kD, 233
 - kD_speed, 234
 - kI, 233
 - kI_nmp, 234
 - kI_speed, 234
 - kP, 233
 - kP_speed, 233
 - kpos_nmp, 234
 - kspeed_nmp, 234
 - maxnpwm, 233
 - maxnpwm_nmp, 234
 - maxppwm, 233
 - maxppwm_nmp, 234
- TMotSFW, 236
- TMotSFW
 - revision, 236
 - subtype, 237
 - subversion, 236
 - type, 236
 - version, 236
- TMotStsFlg
 - kmlMotBase.h, 285
- TMotTPS, 238
- TMotTPS
 - mcfTPS, 238
 - tarpos, 238
- totalTime
 - CLMBase, 89
- TPacket, 239
 - read_sz, 239
 - send_sz, 239
- TPoint3D, 240
 - X, 240
 - Y, 240
 - Z, 240
- TPoint6D, 241
 - Al, 241
 - Be, 241
 - Ga, 241
 - X, 241
 - Y, 241
 - Z, 241
- tps
 - CMotBase, 104
- trajectory
 - CLMBase, 93
- TSctDAT, 243
- TSctDAT
 - arr, 243
 - cnt, 243
- TSctDesc, 244
- TSctDesc
 - ctrlID, 244
 - sens_count, 244
 - sens_res, 244
- TSctGNL, 245
- TSctGNL
 - own, 246
 - res, 246
 - SID, 246
- TSearchDir
 - kmlMotBase.h, 286
- TSplinepoint, 247
 - point, 247
 - time, 247
- type
 - TMotSFW, 236
- uint16
 - CRC.h, 276
- uint8
 - CRC.h, 276
- unBlock
 - CKatana, 72
 - CKatBase, 82
- V1A
 - TBlendtrace, 192
- V1B
 - TBlendtrace, 192
- vel
 - TMotPVP, 230
- ver
 - TKatMFW, 207
- version
 - TMotSFW, 236
- waitForMotor
 - CKatana, 73
 - CMotBase, 102
- WaitParameterException, 252
 - WaitParameterException, 253
- WaitParameterException

- WaitParameterException, [253](#)
- WaitUntilElapsed
 - KNI::Timer, [200](#)
- what
 - Exception, [130](#)
- WriteNotCompleteException, [254](#)
 - WriteNotCompleteException, [255](#)
- WriteNotCompleteException
 - WriteNotCompleteException, [255](#)
- WrongCRCEXception, [257](#)
 - WrongCRCEXception, [258](#)
- WrongCRCEXception
 - WrongCRCEXception, [258](#)
- WrongParameterException, [259](#)
 - WrongParameterException, [260](#)
- WrongParameterException
 - WrongParameterException, [260](#)
- wttc
 - TCdlCOMDesc, [197](#)
- X
 - TPoint3D, [240](#)
 - TPoint6D, [241](#)
- x
 - KNI::KatanaKinematics5M180::position, [142](#)
 - KNI::KatanaKinematics6M180::position, [149](#)
 - KNI::KatanaKinematics6M90G::position, [156](#)
 - KNI::KatanaKinematics6M90T::position, [164](#)
- Y
 - TPoint3D, [240](#)
 - TPoint6D, [241](#)
- y
 - KNI::KatanaKinematics5M180::position, [142](#)
 - KNI::KatanaKinematics6M180::position, [149](#)
 - KNI::KatanaKinematics6M90G::position, [156](#)
 - KNI::KatanaKinematics6M90T::position, [164](#)
- Z
 - TPoint3D, [240](#)
 - TPoint6D, [241](#)
- z
 - KNI::KatanaKinematics5M180::position, [142](#)
 - KNI::KatanaKinematics6M180::position, [149](#)
 - KNI::KatanaKinematics6M90G::position, [156](#)
 - KNI::KatanaKinematics6M90T::position, [164](#)