MOOSGenLib Reference Manual

Generated by Doxygen 1.4.6

Tue Mar 13 10:23:13 2007

Contents

Chapter 1

MOOSGenLib Hierarchical Index

1.1 MOOSGenLib Class Hierarchy

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

Chapter 2

MOOSGenLib Class Index

2.1 MOOSGenLib Class List

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

Chapter 3

MOOSGenLib File Index

3.1 MOOSGenLib File List

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

Chapter 4

MOOSGenLib Class Documentation

4.1 ClosestInterpFunc< T > Class Template Reference

INterfunction for use with InterpBuffer.

`#include <InterpBuffer.h>`

Public Member Functions

- T **operator()** (const val_type &loPair, const val_type &hiPair, double interp_time) const

4.1.1 Detailed Description

template<class T> class ClosestInterpFunc< T >

INterfunction for use with InterpBuffer.

Simply returns closest element to index

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

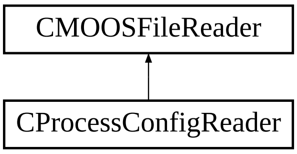
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/InterpBuffer.h

4.2 CMOOSFileReader Class Reference

Base class for reading ascii files.

`#include <MOOSFileReader.h>`

Inheritance diagram for CMOOSFileReader::



Public Member Functions

- bool **IsOpen** ()
- bool **GoTo** (std::string sLine)
- bool **eof** ()
- bool **Reset** ()
- bool **GetValue** (std::string sName, std::string &sResult)
- bool **GetValue** (std::string sName, double &dfResult)
- bool **GetValue** (std::string sName, int &nResult)
- bool **SetFile** (const std::string &sFile)
- std::string **GetNextValidLine** ()

Static Public Member Functions

- static bool **GetTokenValPair** (std::string sLine, std::string &sTok, std::string &sVal)

Protected Member Functions

- std::ifstream * **GetFile** ()

Static Protected Member Functions

- static bool **IsComment** (std::string &sLine)

Protected Attributes

- **CMOOSLock** * **m_pLock**
- std::string **m_sFileName**
- std::ifstream **m_File**
- THREAD2FILE_MAP **m_FileMap**

4.2.1 Detailed Description

Base class for reading ascii files.

4.2.2 Member Function Documentation

4.2.2.1 std::string CMOOSFileReader::GetNextValidLine ()

returns a string of teh next non comment line (and removs trailing comments)

4.2.2.2 bool CMOOSFileReader::GetValue (std::string *sName*, int & *nResult*)

looks for a line "sName = Val" in whole file, fills in result with Val

4.2.2.3 bool CMOOSFileReader::GetValue (std::string *sName*, double & *dfResult*)

looks for a line "sName = Val" in whole file, fills in result with Val

4.2.2.4 bool CMOOSFileReader::GetValue (std::string *sName*, std::string & *sResult*)

looks for a line "sName = Val" in whole file, fills in result with Val

4.2.2.5 bool CMOOSFileReader::SetFile (const std::string & *sFile*)

tell the class what file to read

4.2.3 Member Data Documentation

4.2.3.1 THREAD2FILE_MAP CMOOSFileReader::m_FileMap `[protected]`

every thread get its own pointer to a stream

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSFileReader.h
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSFileReader.cpp

4.3 CMOOSGeodesy Class Reference

Implements simple geodesy calculations.

`#include <MOOSGeodesy.h>`

Public Member Functions

- double **GetOriginNorthing** ()
- double **GetOriginEasting** ()
- bool **LatLong2LocalUTM** (double lat, double lon, double &MetersNorth, double &MetersEast)
- char ∗ **GetUTMZone** ()
- int **GetRefEllipsoid** ()
- double **GetMetersEast** ()
- double **GetMetersNorth** ()
- double **GetOriginLatitude** ()
- double **GetOriginLongitude** ()
- bool **Initialise** (double lat, double lon)
- double **GetLocalGridY** ()
- double **GetLocalGridX** ()
- bool **LatLong2LocalGrid** (double lat, double lon, double &MetersNorth, double &MetersEast)
- double **DMS2DecDeg** (double dfVal)

Classes

- class **CEllipsoid**

4.3.1 Detailed Description

Implements simple geodesy calculations.

4.3.2 Member Function Documentation

4.3.2.1 bool CMOOSGeodesy::Initialise (double *lat*, double *lon*)

This method is called to set the Origins of the Coordinate system being used by the vehicle for a mission. This class will store the vehicle's position in Northings and Eastings. This allows for tracking the vehicle as if it were operating on a grid.

Parameters:

lat the Latitude of where the vehicle is as it begins a mission

lon the Longitude of where the vehicle is as it begins a mission

Returns:

only returns true at the moment, no reason as to why it should fail perhaps some way of checking UTM zones vs a list of some sort?

4.3.2.2 bool CMOOSGeodesy::LatLong2LocalUTM (double *lat*, double *lon*, double & *MetersNorth*, double & *MetersEast*)

This method is the interface to this class and allows the client to query the amount of ground covered with respect to the origin where the origin is defined as a point in the UTM grid where we got an initial GPS fix that we defined to be the origin. What this method does not take into account is the curvature of the reference ellipsoid at a particular Lat/Lon value. Curvature influences the deltaX and deltaY that this method calculates for determining the overall distance traveled wrt the origin. Therefore, at Lat/Lon values that are significantly far enough (~300km) away from the origin of the UTM grid (0,0), a shift in one dimension, i.e. just along Latitude, or just along Longitude, does not map to a corresponding one dimensional shift in our "local" grid where we should be seeing just a deltaX or deltaY result in moving in only one direction. Instead, we have observed that moving just .0001 degrees in Longitude (~1m in local) results in both a deltaX that is coupled to a deltaY.

Parameters:

lat The current Latitude the vehicle is at

lon The current Longitude the vehicle is at

MetersNorth The distance in meters traveled North wrt to Origin

MetersEast The distance in meters traveled East wrt to Origin

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

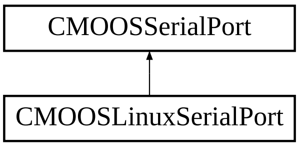
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSGeodesy.h
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSGeodesy.cpp

4.4 CMOOSLinuxSerialPort Class Reference

Implements linux aspects of **CMOOSSerialPort**(p. 21).

`#include <MOOSLinuxSerialPort.h>`

Inheritance diagram for CMOOSLinuxSerialPort::



Public Member Functions

- virtual bool **Close** ()
- **CMOOSLinuxSerialPort** ()
- virtual ~**CMOOSLinuxSerialPort** ()
- virtual bool **Create** (const char *pPortNum=DEFAULT_PORT, int nBaudRate=DEFAULT_BAUDRATE)
- int **Write** (char *Str, int nLen, double *pTime=NULL)
- virtual void **Break** ()
- virtual int **Flush** ()
- int **GetFD** ()

Protected Member Functions

- virtual int **GrabN** (char *pBuffer, int nRequired)

Protected Attributes

- int **m_nPortFD**
- termios **m_OldPortOptions**
- termios **m_PortOptions**

4.4.1 Detailed Description

Implements linux aspects of **CMOOSSerialPort**(p. 21).

4.4.2 Constructor & Destructor Documentation

4.4.2.1 CMOOSLinuxSerialPort::CMOOSLinuxSerialPort ()

constructor.

4.4.2.2 CMOOSLinuxSerialPort::~CMOOSLinuxSerialPort () `[virtual]`

Destructor.Reset the port option to what every they were before and close port

4.4.3 Member Function Documentation

4.4.3.1 void CMOOSLinuxSerialPort::Break () `[virtual]`

send break signal

Reimplemented from **CMOOSSerialPort** (p. 21).

4.4.3.2 bool CMOOSLinuxSerialPort::Create (const char * *pPortNum* = `DEFAULT_PORT`, int *nBaudRate* = `DEFAULT_BAUDRATE`) `[virtual]`

Create and set up the port

Implements **CMOOSSerialPort** (p. 21).

4.4.3.3 int CMOOSLinuxSerialPort::Flush (void) `[virtual]`

Call this method in order to free the Output Buffer of any characters that may not have been sent during our last write. We use the queue_selector TCOFLUSH.

See also:

`http://www.mkssoftware.com/docs/man3/tcflush.3.asp`

Reimplemented from **CMOOSSerialPort** (p. 21).

4.4.3.4 int CMOOSLinuxSerialPort::GetFD ()

returns the file descriptor

4.4.3.5 int CMOOSLinuxSerialPort::GrabN (char * *pBuffer*, int *nRequired*) `[protected, virtual]`

Just grab N characters NOW

Implements **CMOOSSerialPort** (p. 21).

4.4.3.6 int CMOOSLinuxSerialPort::Write (char * *Str*, int *nLen*, double * *pTime* = `NULL`) `[virtual]`

Write a string out of port

Implements **CMOOSSerialPort** (p. 21).

4.4.4 Member Data Documentation

4.4.4.1 int CMOOSLinuxSerialPort::m_nPortFD `[protected]`

FileDescriptor of Port

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSLinuxSerialPort.h
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSLinuxSerialPort.cpp

4.5 CMOOSLock Class Reference

`#include <MOOSLock.h>`

Public Member Functions

- void **UnLock** ()
  call this to unlock
- void **Lock** ()
  call this to lock
- **CMOOSLock** (bool bInitial=true)

Protected Attributes

- pthread_mutex_t **m_hLock**
  posix mutex

4.5.1 Detailed Description

A very simple cross platform posix and win32 compatible mutex class

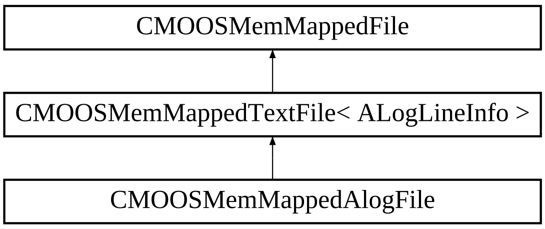
The documentation for this class was generated from the following files:

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSLock.h
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSLock.cpp

4.6 CMOOSMemMappedAlogFile Class Reference

`#include <MOOSMemoryMapped.h>`

Inheritance diagram for CMOOSMemMappedAlogFile::



Public Member Functions

- bool **Open** (const std::string &sName, bool bSummary=true, int nMaxLines=-1)
- bool **SortLineIndex** ()
- int **SeekToFindTime** (double dfT)
- bool **ReadSourceAndTypeSets** ()
- bool **ReadStartTime** ()
- double **GetLogStart** ()
- double **GetEntryTime** (int i)
- std::set< std::string > **GetSourceNames** ()
- std::set< std::string > **GetMessageNames** ()

Static Public Member Functions

- static bool **TimePredicate** (const **LINE** &L, double dfT)
- static bool **TimePredicate2** (const **LINE** &L, const **LINE** &L2)
- static bool **GetNextToken** (const std::string &s, int &nPos, std::string &sTk)

Public Attributes

- std::set< std::string > **m_MessageNames**
- std::set< std::string > **m_SourceNames**
- double **m_dfLogStart**

4.6.1 Detailed Description

specialisation of memory mapped ASCII file to swoop around MOOS alog files. Main specialisation here is that the templated type is a ALogLineInfo struct which contains time information. This is sorted during creation

4.6.2 Member Function Documentation

4.6.2.1 bool CMOOSMemMappedAlogFile::ReadSourceAndTypeSets () `[inline]`

This function builds two sets , one conataining all the unique messgae names and the other containing the set of message sources (processes)

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSMemoryMapped.h

4.7 CMOOSMemMappedTextFile< T > Class Template Reference

`#include <MOOSMemoryMapped.h>`

Public Types

- typedef T **LINE**

Public Member Functions

- **CMOOSMemMappedTextFile** ()
- bool **Open** (const std::string &sName, int nMaxLines=-1)
- int **GetLineCount** ()
- bool **BuildLineIndex** (int nNumLines=-1)
- std::string **GetLine** (int nLine)

Public Attributes

- std::vector< **LINE** > **m_LineIndex**
- int **m_nLineCount**

4.7.1 Detailed Description

template<class T = TextLineInfo> class CMOOSMemMappedTextFile< T >

Specialised memory mapped file class to handle text files.

Default template parameter is type **TextLineInfo**(p. 35). Other template types must support the following functions: T::T(char * pStart, char * pEnd) //constructor char * T.Start(); //return start of line char * T.End(); //return end of line bool IsWanted(); //return true if this line is wanted

Main functionality of this class is to produce a index of lines each entry pointing to a instance of "T"

4.7.2 Member Typedef Documentation

4.7.2.1 template<class T = TextLineInfo> typedef T CMOOSMemMappedTextFile< T >::LINE

we will refer to T as type LINE

4.7.3 Constructor & Destructor Documentation

4.7.3.1 template<class T = TextLineInfo> CMOOSMemMappedTextFile< T >::CMOOSMemMappedTextFile () `[inline]`

constructor counts lines and buld line index

4.7.4 Member Function Documentation

4.7.4.1 template<class T = TextLineInfo> bool CMOOSMemMappedTextFile< T >::BuildLineIndex (int *nNumLines* = -1) [inline]

iterate through the file building the line index

4.7.4.2 template<class T = TextLineInfo> std::string CMOOSMemMappedTextFile< T >::GetLine (int *nLine*) [inline]

return the specified line as a string

4.7.4.3 template<class T = TextLineInfo> int CMOOSMemMappedTextFile< T >::GetLineCount () [inline]

simply returns size of file in lines looking for

4.7.5 Member Data Documentation

4.7.5.1 template<class T = TextLineInfo> std::vector< LINE > CMOOSMemMappedTextFile< T >::m_LineIndex

a vector of LINES (templated type).

4.7.5.2 template<class T = TextLineInfo> int CMOOSMemMappedTextFile< T >::m_nLineCount

how many lines we have

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

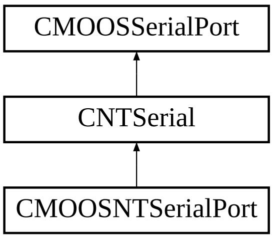
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSMemoryMapped.h

4.8 CMOOSNTSerialPort Class Reference

Implements windows specfialisations of MOOSSerialPort.

`#include <MOOSNTSerialPort.h>`

Inheritance diagram for CMOOSNTSerialPort::



Public Member Functions

- virtual void **Break** ()
- virtual bool **Create** (const char ∗pPortNum=DEFAULT_PORT, int nBaudRate=DEFAULT_BAUDRATE)
- bool **Close** (void)
- int **Write** (char ∗pData, int nLen, double ∗pTime=NULL)

Protected Member Functions

- virtual int **GrabN** (char ∗pBuffer, int nRequired)

4.8.1 Detailed Description

Implements windows specfialisations of MOOSSerialPort.

4.8.2 Member Function Documentation

4.8.2.1 void CMOOSNTSerialPort::Break () `[virtual]`

Send break signal

Reimplemented from **CMOOSSerialPort** (p. 21).

4.8.2.2 bool CMOOSNTSerialPort::Close (void) `[virtual]`

Close Port

Reimplemented from **CMOOSSerialPort** (p. 21).

4.8.2.3 bool CMOOSNTSerialPort::Create (const char ∗ *pPortNum* = `DEFAULT_PORT`, int *nBaudRate* = `DEFAULT_BAUDRATE`) `[virtual]`

Create an open port

Implements **CMOOSSerialPort** (p. 21).

4.8.2.4 int CMOOSNTSerialPort::GrabN (char ∗ *pBuffer*, int *nRequired*) `[protected, virtual]`

Grab N chars NOW

Implements **CMOOSSerialPort** (p. 21).

4.8.2.5 int CMOOSNTSerialPort::Write (char ∗ *pData*, int *nLen*, double ∗ *pTime* = `NULL`) `[virtual]`

Write nLen bytes out

Implements **CMOOSSerialPort** (p. 21).

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

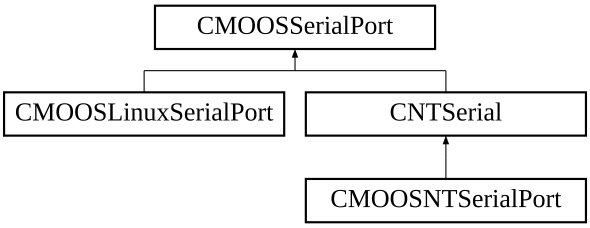
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSNTSerialPort.h
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSNTSerialPort.cpp

4.9 CMOOSSerialPort Class Reference

Cross Platform Serial Port Base Class.

`#include <MOOSSerialPort.h>`

Inheritance diagram for CMOOSSerialPort::



Public Types

- typedef std::list< CMOOSSerialTelegram > **TELEGRAM_LIST**

Public Member Functions

- std::string **GetPortName** ()
- virtual bool **Close** ()
- char **GetTermCharacter** ()
- void **SetTermCharacter** (char cTermChar)
- int **GetBaudRate** ()
- virtual int **Flush** ()
- bool **IsStreaming** ()
- bool **IsVerbose** ()
- bool **GetLatest** (std::string &sWhat, double &dfWhen)
- bool **GetEarliest** (std::string &sWhat, double &dfWhen)
- bool **CommsLoop** ()
- virtual bool **Configure** (STRING_LIST sParams)
- virtual bool **Create** (const char ∗pPortNum=DEFAULT_PORT, int nBaudRate=DEFAULT_BAUDRATE)=0
- virtual int **ReadNWithTimeOut** (char ∗pBuff, int nBufferLen, double Timeout=0.5, double ∗pTime=NULL)
- virtual int **Write** (char ∗Str, int nLen, double ∗pTime=NULL)=0
- bool **GetTelegram** (std::string &sTelegram, double dfTimeOut, double ∗pTime=NULL)
- void **SetIsCompleteReplyCallBack** (bool(∗pfn)(char ∗pData, int nLen, int nRead))
- virtual void **Break** ()

Public Attributes

- TELEGRAM_LIST **m_InBox**
- TELEGRAM_LIST **m_OutBox**
- **CMOOSLock m_InBoxLock**
- **CMOOSLock m_OutBoxLock**
- **CMOOSLock m_PortLock**

Protected Types

- typedef pthread_t **THREAD_ID**

Protected Member Functions

- bool **StartThreads** ()
- virtual int **GrabN** (char ∗pBuffer, int nRequired)=0
- bool **IsCompleteReply** (char ∗pData, int nLen, int nRead)

Protected Attributes

- char **m_cTermCharacter**
- **THREAD_ID m_nCommsThreadID**
- bool **m_bStreaming**
- bool **m_bVerbose**
- bool(∗ **m_pfnUserIsCompleteReplyCallBack**)(char ∗pData, int nLen, int nRead)
- bool **m_bHandShaking**

  handware handshaking active flag

- std::string **m_sPort**

  port name

- int **m_nBaudRate**

  baudrate

- bool **m_bQuit**
- bool **m_bUseCsmExt**

  ARH 14/05/2005 For 500kBaud PCMCIA card.

Classes

- class **CMOOSSerialTelegram**

4.9.1 Detailed Description

Cross Platform Serial Port Base Class.

Provides cross platform functionality which is implemented in detail by the platform dependent derivatives

4.9.2 Member Typedef Documentation

4.9.2.1 typedef pthread_t CMOOSSerialPort::THREAD_ID `[protected]`

Win32 handle to IO thread

4.9.3 Member Function Documentation

4.9.3.1 void CMOOSSerialPort::SetTermCharacter (char *cTermChar*)

Sets the termination character for the serial port to watch out for when it constructs Telegrams for Streaming Devices.

4.9.4 Member Data Documentation

4.9.4.1 THREAD_ID CMOOSSerialPort::m_nCommsThreadID `[protected]`

ID of IO thread

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/**MOOSSerialPort.h**
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSSerialPort.cpp

4.10 CMOOSThread Class Reference

Implements a cross platform thread*/.

`#include <MOOSThread.h>`

Public Member Functions

- **CMOOSThread** (t_pfnWorkerFunc pfnThreadFunc, void *pThreadData)
- ~**CMOOSThread** ()

  Destructor just stops the thread if there's one running.

- bool **Initialise** (t_pfnWorkerFunc pfnThreadFunc, void *pThreadData)
- bool **IsQuitRequested** ()
- bool **IsThreadRunning** ()
- bool **Start** ()

  Starts the thread running (as long as the class has been properly initialised!).

- bool **Stop** ()

Static Public Member Functions

- static TCB **CallbackProc** (void *lpThis)

4.10.1 Detailed Description

Implements a cross platform thread*/.

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

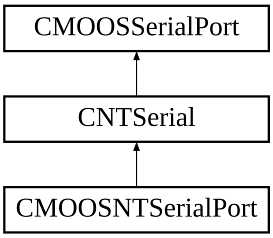
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSThread.h

4.11 CNTSerial Class Reference

Middle Layer class for Windows Serial port (c) Ramon de Klein.

`#include <NTSerial.h>`

Inheritance diagram for CNTSerial::



Public Types

- enum **EEvent** {

  EEventNone = -1, **EEventBreak** = EV_BREAK, **EEventCTS** = EV_CTS, **EEventDSR** = EV_DSR,

  EEventError = EV_ERR, **EEventRing** = EV_RING, **EEventRLSD** = EV_RLSD, **EEventRecv** = EV_RXCHAR,

  EEventRcvEv = EV_RXFLAG, **EEventSend** = EV_TXEMPTY }
- enum **EBaudrate** {

  EBaudUnknown = -1, **EBaud110** = CBR_110, **EBaud300** = CBR_300, **EBaud600** = CBR_600,

  EBaud1200 = CBR_1200, **EBaud2400** = CBR_2400, **EBaud4800** = CBR_4800, **EBaud9600** = CBR_9600,

  EBaud14400 = CBR_14400, **EBaud19200** = CBR_19200, **EBaud38400** = CBR_38400, **EBaud56000** = CBR_56000,

  EBaud57600 = CBR_57600, **EBaud115200** = CBR_115200, **EBaud128000** = CBR_128000, **EBaud256000** = CBR_256000,

  EBaud500000 = 500000, **EBaudCSM9600** = 2150, **EBaudCSM19200** = 4301, **EBaudCSM38400** = 8602,

  EBaudCSM500000 = 115000 }
- enum **EDataBits** {

  EDataUnknown = -1, **EData5** = 5, **EData6** = 6, **EData7** = 7,

  EData8 = 8 }
- enum **EParity** {

  EParUnknown = -1, **EParNone** = NOPARITY, **EParOdd** = ODDPARITY, **EParEven** = EVENPARITY,

  EParMark = MARKPARITY, **EParSpace** = SPACEPARITY }
- enum **EStopBits** { **EStopUnknown** = -1, **EStop1** = ONESTOPBIT, **EStop1_5** = ONE5STOPBITS, **EStop2** = TWOSTOPBITS }
- enum **EHandshake** { **EHandshakeUnknown** = -1, **EHandshakeOff** = 0, **EHandshakeHardware** = 1, **EHandshakeSoftware** = 2 }

- enum **EReadTimeout** { **EReadTimeoutUnknown** = -1, **EReadTimeoutNonblocking** = 0, **EReadTimeoutBlocking** = 1 }
- enum **EError** {

  EErrorUnknown = 0, **EErrorBreak** = CE_BREAK, **EErrorFrame** = CE_FRAME, **EErrorIOE** = CE_IOE,

  EErrorMode = CE_MODE, **EErrorOverrun** = CE_OVERRUN, **EErrorRxOver** = CE_RXOVER, **EErrorParity** = CE_RXPARITY,

  EErrorTxFull = CE_TXFULL }
- enum **EPort** { **EPortUnknownError** = -1, **EPortAvailable** = 0, **EPortNotAvailable** = 1, **EPortInUse** = 2 }

Public Member Functions

- EPort **CheckPort** (LPCTSTR lpszDevice)
- virtual LONG **Open** (LPCTSTR lpszDevice, DWORD dwInQueue=2048, DWORD dwOutQueue=2048)
- virtual LONG **ClosePort** (void)
- virtual LONG **Setup** (EBaudrate eBaudrate=EBaud9600, EDataBits eDataBits=EData8, EParity eParity=EParNone, EStopBits eStopBits=EStop1)
- virtual LONG **SetEventChar** (BYTE bEventChar, bool fAdjustMask=true)
- virtual LONG **SetMask** (DWORD dwMask=EEventBreak|EEventError|EEventRecv)
- virtual LONG **WaitEvent** (LPOVERLAPPED lpOverlapped=0, DWORD dwTimeout=INFINITE)
- virtual LONG **SetupHandshaking** (EHandshake eHandshake)
- virtual LONG **SetupReadTimeouts** (EReadTimeout eReadTimeout)
- virtual EBaudrate **GetBaudrate** (void)
- virtual EDataBits **GetDataBits** (void)
- virtual EParity **GetParity** (void)
- virtual EStopBits **GetStopBits** (void)
- virtual EHandshake **GetHandshaking** (void)
- virtual DWORD **GetEventMask** (void)
- virtual BYTE **GetEventChar** (void)
- virtual LONG **Write** (const void ∗pData, size_t iLen, DWORD ∗pdwWritten=0, LPOVERLAPPED lpOverlapped=0, DWORD dwTimeout=INFINITE)
- virtual LONG **Write** (LPCSTR pString, DWORD ∗pdwWritten=0, LPOVERLAPPED lpOverlapped=0, DWORD dwTimeout=INFINITE)
- virtual LONG **NTRead** (void ∗pData, size_t iLen, DWORD ∗pdwRead=0, LPOVERLAPPED lpOverlapped=0, DWORD dwTimeout=INFINITE)
- EEvent **GetEventType** (void)
- EError **GetError** (void)
- HANDLE **GetCommHandle** (void)
- bool **IsOpen** (void) const
- LONG **GetLastError** (void) const
- bool **GetCTS** (void)
- bool **GetDSR** (void)
- bool **GetRing** (void)
- bool **GetRLSD** (void)
- virtual int **Flush** (void)

Protected Attributes

- LONG **m_lLastError**
- HANDLE **m_hFile**
- EEvent **m_eEvent**
- HANDLE **m_hevtOverlapped**

Classes

- class **CDCB**

4.11.1 Detailed Description

Middle Layer class for Windows Serial port (c) Ramon de Klein.

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

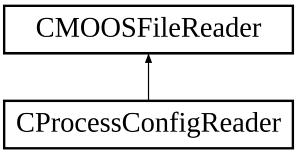
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/NTSerial.h
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/NTSerial.cpp

4.12 CProcessConfigReader Class Reference

Class for reading MOOS configuration files.

`#include <ProcessConfigReader.h>`

Inheritance diagram for CProcessConfigReader::



Public Member Functions

- std::string **GetAppName** ()
- std::string **GetFileName** ()
- void **SetAppName** (std::string sAppName)
- bool **GetConfigurationParam** (std::string sAppName, std::string sParam, std::string &sVal)

  READ STRINGS.

- bool **GetConfigurationParam** (std::string sAppName, std::string sParam, double &dfVal)
- bool **GetConfigurationParam** (std::string sAppName, std::string sParam, bool &bVal)
- bool **GetConfigurationParam** (std::string sAppName, std::string sParam, int &nVal)
- bool **GetConfigurationParam** (std::string sAppName, std::string sParam, std::vector< double > &Vec, int &nRows, int &nCols)
- bool **GetConfigurationParam** (std::string sParam, std::string &sVal)
- bool **GetConfigurationParam** (std::string sParam, double &dfVal)

  READ DOUBLES.

- bool **GetConfigurationParam** (std::string sParam, bool &bVal)

  READ BOOLS.

- bool **GetConfigurationParam** (std::string sParam, int &nVal)

  READ INTS.

- bool **GetConfigurationParam** (std::string sParam, std::vector< double > &Vec, int &nRows, int &nCols)

  READ VECTORS.

- bool **GetConfiguration** (std::string sAppName, STRING_LIST &Params)

Public Attributes

- std::string **m_sAppName**

4.12.1 Detailed Description

Class for reading MOOS configuration files.

4.12.2 Member Function Documentation

4.12.2.1 std::string CProcessConfigReader::GetAppName ()

returns the name of the application an instance of this class is concerned with

4.12.2.2 bool CProcessConfigReader::GetConfiguration (std::string *sAppName*, STRING_LIST & *Params*)

return a list of strings of Token = Val for the specfied named application configuration block

4.12.2.3 bool CProcessConfigReader::GetConfigurationParam (std::string *sParam*, std::vector< double > & *Vec*, int & *nRows*, int & *nCols*)

READ VECTORS.

read a vector<double> parameter for a Process "m_sName" (can be interprested as a matrix with (rows x cols)

4.12.2.4 bool CProcessConfigReader::GetConfigurationParam (std::string *sParam*, int & *nVal*)

READ INTS.

read a int parameter for a Process "m_sName"

4.12.2.5 bool CProcessConfigReader::GetConfigurationParam (std::string *sParam*, bool & *bVal*)

READ BOOLS.

read a bool parameter for a Process "m_sName"

4.12.2.6 bool CProcessConfigReader::GetConfigurationParam (std::string *sParam*, double & *dfVal*)

READ DOUBLES.

read a double parameter for a Process "m_sName"

4.12.2.7 bool CProcessConfigReader::GetConfigurationParam (std::string *sParam*, std::string & *sVal*)

read a string parameter for a Process "m_sName"

4.12.2.8 bool CProcessConfigReader::GetConfigurationParam (std::string *sAppName*, std::string *sParam*, std::vector< double > & *Vec*, int & *nRows*, int & *nCols*)

read a vector<double> parameter for a named process

4.12.2.9 bool CProcessConfigReader::GetConfigurationParam (std::string *sAppName*, std::string *sParam*, int & *nVal*)

read a integer parameter for a named process

4.12.2.10 bool CProcessConfigReader::GetConfigurationParam (std::string *sAppName*, std::string *sParam*, bool & *bVal*)

read a bool parameter for a named process

4.12.2.11 bool CProcessConfigReader::GetConfigurationParam (std::string *sAppName*, std::string *sParam*, double & *dfVal*)

read a string parameter for a named process

4.12.2.12 bool CProcessConfigReader::GetConfigurationParam (std::string *sAppName*, std::string *sParam*, std::string & *sVal*)

READ STRINGS.

read a string parameter for a named process

4.12.2.13 std::string CProcessConfigReader::GetFileName ()

returns the name of the mission file this process is accessing

4.12.2.14 void CProcessConfigReader::SetAppName (std::string *sAppName*)

set the name of the application (MOOSProcess) that this class shoud concern itself with (unless directed otherwise)

4.12.3 Member Data Documentation

4.12.3.1 std::string CProcessConfigReader::m_sAppName

the name of process an instance this class will handle unless told otherwise

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/**ProcessConfigReader.h**
- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/ProcessConfigReader.cpp

4.13 CTimeGenericInterpolator< T > Class Template Reference

TimeInterpolator for use with InterpBuffer.

`#include <InterpBuffer.h>`

Public Types

- typedef std::pair< double, T > **TIME_DOUBLE_VAL_PAIR**

Public Member Functions

- T **operator()** (const TIME_DOUBLE_VAL_PAIR &loPair, const TIME_DOUBLE_VAL_PAIR &hiPair, double dfInterpTime) const

4.13.1 Detailed Description

template<class T> class CTimeGenericInterpolator< T >

TimeInterpolator for use with InterpBuffer.

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/InterpBuffer.h

4.14 CTimeNumericInterpolator< T > Class Template Reference

Interpolator to use with interpbuffer.

`#include <InterpBuffer.h>`

Public Types

- typedef std::pair< double, T > **TIME_DOUBLE_NUM_PAIR**

Public Member Functions

- T **operator()** (const TIME_DOUBLE_NUM_PAIR &loPair, const TIME_DOUBLE_NUM_PAIR &hiPair, double dfInterpTime) const

4.14.1 Detailed Description

template<class T> class CTimeNumericInterpolator< T >

Interpolator to use with interpbuffer.

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/InterpBuffer.h

4.15 dynamic_caster< D > Struct Template Reference

`#include <MOOSGenLibGlobalHelper.h>`

Public Member Functions

- template<class S> D **operator()** (S s) const

4.15.1 Detailed Description

template<class D> struct dynamic_caster< D >

Functor class for performing dynamic_cast between two types. Use it with stl::transform when copying between two collections with different element types

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/**MOOSGenLibGlobalHelper.h**

4.16 static_caster< D > Struct Template Reference

`#include <MOOSGenLibGlobalHelper.h>`

Public Member Functions

- template<class S> D **operator()** (S s) const

4.16.1 Detailed Description

template<class D> struct static_caster< D >

Functor class for performing static_cast between two types. Use it with stl::transform when copying between two collections with different element types

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/**MOOSGenLibGlobalHelper.h**

4.17 TextLineInfo Struct Reference

`#include <MOOSMemoryMapped.h>`

Public Member Functions

- **TextLineInfo** (char *pStart, char *pEnd)
- char * **Start** ()
- char * **End** ()
- bool **IsWanted** ()

Public Attributes

- char * **pLineStart**
- char * **pLineEnd**

4.17.1 Detailed Description

simple structure to hold text line info

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSMemoryMapped.h

4.18 TInterpBuffer< Key, Data, InterpFunc, Compare > Class Template Reference

a buffer to store data and get interp values by index with time

`#include <InterpBuffer.h>`

Public Member Functions

- void **SetInterpFunc** (const InterpFunc &interp)
- InterpFunc & **GetInterpFunc** ()
- Data **operator()** (const Key &interp_time) const
- void **MakeSpanTime** (double dfSpan)
- void **EraseOld** (double dfTime)
- Key **MaxKey** () const
- Key **MinKey** () const
- bool **MaxData** (Data &D) const

4.18.1 Detailed Description

template<class Key, class Data, class InterpFunc, class Compare = std::less<Key>> class TInterpBuffer< Key, Data, InterpFunc, Compare >

a buffer to store data and get interp values by index with time

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

- /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/InterpBuffer.h

Chapter 5

MOOSGenLib File Documentation

5.1 /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSGenLibGlobalHelper.h File Reference

`#include <string>`

`#include <list>`

`#include <vector>`

`#include <algorithm>`

Classes

- struct **static_caster**< **D** >
- struct **dynamic_caster**< **D** >

Defines

- #define **MOOSHERE** MOOSFormat("File %s Line %d", __FILE__,__LINE__).c_str()

Typedefs

- typedef std::list< std::string > **STRING_LIST**

Functions

- bool **MOOSGetValueFromToken** (STRING_LIST &sParams, const std::string &sToken, std::string &sVal)
- bool **MOOSValFromString** (std::string &sVal, const std::string &sStr, const std::string &sTk)
- bool **MOOSValFromString** (double &dfVal, const std::string &sStr, const std::string &sTk)
- bool **MOOSValFromString** (int &nVal, const std::string &sStr, const std::string &sTk)

- bool **MOOSValFromString** (std::vector< double > &dfValVec, int &nRows, int &nCols, const std::string &sStr, const std::string &sToken)
- bool **MOOSVectorFromString** (const std::string &sStr, std::vector< double > &dfVecVal, int &nRows, int &nCols)
- std::string **DoubleVector2String** (const std::vector< double > &V)
- std::stringstream & **Write** (std::stringstream &os, const std::vector< double > &Vec)
- std::stringstream & **Write** (std::stringstream &os, const std::vector< int > &Vec)
- std::string **MOOSChomp** (std::string &sStr, const std::string &sTk=",")
- void **MOOSRemoveChars** (std::string &sStr, const std::string &sTok)
- void **MOOSToUpper** (std::string &str)
- void **MOOSTrimWhiteSpace** (std::string &str)
- bool **MOOSIsNumeric** (std::string str)
- bool **MOOSStrCmp** (std::string s1, std::string s2)
- void **SetMOOSSkew** (double dfSkew)
- void **MOOSPause** (int nMS)
- double **MOOSTime** ()
- double **HPMOOSTime** ()
- int **MOOSGetch** ()
- void **MOOSTrace** (std::string Str)
- void **MOOSTrace** (char ∗FmtStr,...)
- std::string **MOOSFormat** (char ∗FmtStr,...)
- bool **MOOSFail** (char ∗FmtStr,...)
- std::string **MOOSGetTimeStampString** ()
- std::string **MOOSGetDate** ()
- void **Progress** (double dfPC)
- std::string **MOOSThirdPartyActuationString** (double ∗pdfRudder, double ∗pdfElevator, double ∗pdfThrust)
- std::string **MOOSThirdPartyStatusString** (std::string sStatusCommand)
- double **MOOS_ANGLE_WRAP** (double dfAng)
- double **MOOSDeg2Rad** (double dfDeg)
- double **MOOSRad2Deg** (double dfRad)
- bool **MOOSAbsLimit** (double &dfVal, double dfLimit)
- double **MOOSWhiteNoise** (double Sigma)
- double **MOOSNormalInv** (double dfArea)
- int **MOOSDiscreteUniform** (int nMin, int nMax)
- double **MOOSUniformRandom** (double dfMin, double dfMax)
- template<class T> const T & **MOOSClamp** (const T &val, const T &min, const T &max)
- bool **GetDirectoryContents** (const std::string &sPath, std::list< std::string > &sContents, bool bFiles=true)
- bool **MOOSCreateDirectory** (const std::string &sDirectory)
- template<class T> T **SwapByteOrder** (const T &v)
- bool **IsLittleEndian** ()

5.1.1 Detailed Description

5.1.2 Define Documentation

5.1.2.1 #define MOOSHERE MOOSFormat("File %s Line %d", __FILE__,__LINE__).c_str()

useful macro for debugging prints line and file

5.1.3 Function Documentation

5.1.3.1 std::string DoubleVector2String (const std::vector< double > & *V*)

formats a vector of doublse into standard MOOS format

5.1.3.2 bool GetDirectoryContents (const std::string & *sPath*, std::list< std::string > & *sContents*, bool *bFiles*)

returns a string list of directories or files in a specified location exludes . and ..

5.1.3.3 double HPMOOSTime ()

return high precision timestamp - time since unix in seconds

5.1.3.4 bool IsLittleEndian ()

returns true if architecture is LittleEndian (true for x86 Architectures) Note after first call it remembers answer in a static so v. littel overhead in calling this function frequently

5.1.3.5 double MOOS_ANGLE_WRAP (double *dfAng*)

Bound angle to +/-PI

5.1.3.6 bool MOOSAbsLimit (double & *dfVal*, double *dfLimit*)

limit dfVal to lie between +/- dfLimit)

5.1.3.7 template<class T> const T& MOOSClamp (const T & *val*, const T & *min*, const T & *max*)

Clamps a templated type between two values

5.1.3.8 bool MOOSCreateDirectory (const std::string & *sDirectory*)

make a directory

5.1.3.9 double MOOSDeg2Rad (double *dfDeg*)

convert deg to rad

5.1.3.10 int MOOSDiscreteUniform (int *nMin*, int *nMax*)

generates uniform noise in integers between interval nMin->nMax

5.1.3.11 bool MOOSFail (char * *FmtStr*, ...)

like MOOSTrace but returns false - useful for return statements

5.1.3.12 std::string MOOSFormat (char ∗ *FmtStr*, *...*)

return a formatted string (with printf-like format codes

5.1.3.13 int MOOSGetch ()

useful keyboard trap

5.1.3.14 std::string MOOSGetDate ()

get teh current date formatted nicely

5.1.3.15 std::string MOOSGetTimeStampString ()

return nicely formatted time stamp string

5.1.3.16 bool MOOSIsNumeric (std::string *str*)

returbn true if numeric

5.1.3.17 double MOOSNormalInv (double *dfArea*)

returns x for probablity mass such p(v<=x) = dfArea)

5.1.3.18 void MOOSPause (int *nMS*)

pause for nMS milliseconds

5.1.3.19 double MOOSRad2Deg (double *dfRad*)

convert rad 2 deg

5.1.3.20 void MOOSRemoveChars (std::string & *sStr*, const std::string & *sTok*)

remove all characters in sTok from sStr

5.1.3.21 bool MOOSStrCmp (std::string *s1*, std::string *s2*)

case insensitive string comparison. returns true if equal

5.1.3.22 double MOOSTime ()

return time as a double (time since unix in seconds)

5.1.3.23 void MOOSToUpper (std::string & *str*)

convert string to upper case

5.1.3.24 void MOOSTrace (char ∗ *FmtStr*, *...*)

print a formatted string (with printf-like format codes) and to debug window in DevStudio

5.1.3.25 void MOOSTrace (std::string *Str*)

print a string

5.1.3.26 void MOOSTrimWhiteSpace (std::string & *str*)

remove white space form start and end of a string

5.1.3.27 double MOOSUniformRandom (double *dfMin*, double *dfMax*)

generates uniform noise in interval dfMin-dfMax

5.1.3.28 double MOOSWhiteNoise (double *Sigma*)

returns sample fom Gaussian process strength Sigma mena zero

5.1.3.29 void Progress (double *dfPC*)

prints a "progress bar" upto 40 characters long dfPC is the fraction complete - ie 0:1

5.1.3.30 void SetMOOSSkew (double *dfSkew*)

generic timing functions

5.1.3.31 template<class T> T SwapByteOrder (const T & *v*)

templated function which swaps byte order of type T returning it

5.1.3.32 std::stringstream& Write (std::stringstream & *os*, const std::vector< int > & *Vec*)

formats a vector of ints into standard MOOS format

5.1.3.33 std::stringstream& Write (std::stringstream & *os*, const std::vector< double > & *Vec*)

formats a vector of doubles into standard MOOS format

5.2 /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSSerialPort.h File Reference

`#include "MOOSLock.h"`

`#include <string>`

`#include <list>`

Classes

- class **CMOOSSerialPort**

  Cross Platform Serial Port Base Class.

- class **CMOOSSerialPort::CMOOSSerialTelegram**

Defines

- #define **MOOSSERIALPORTH**
- #define **DEFAULT_PORT** "/dev/ttyS0"
- #define **DEFAULT_BAUDRATE** 9600
- #define **TELEGRAM_LEN** 1000

Typedefs

- typedef std::list< std::string > **STRING_LIST**

5.2.1 Detailed Description

5.3 /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/MOOSTimeJournal.h File Reference

```
#include <fstream>
#include <map>
#include <string>
```

Classes

- class **CMOOSTimeJournal**

5.3.1 Detailed Description

5.4 /home/pnewman/code/MOOS/trunk/Core/MOOSGenLib/ProcessConfigReader.h File Reference

```
#include "MOOSFileReader.h"
#include <string>
#include <list>
#include <vector>
```

Classes

- class **CProcessConfigReader**
  Class for reading MOOS configuration files.

Typedefs

- typedef std::list< std::string > **STRING_LIST**

5.4.1 Detailed Description

Index