MAMA C++ API Reference Manual

2.2.2.1

Generated by Doxygen 1.4.7

Thu Feb 7 17:04:48 2013

Contents

Chapter 1

Middleware Agnostic Messaging API (MAMA) C++ API

Middleware Agnostic Messaging API. The Middleware Agnostic Messaging (MAMA) API provides an abstraction layer over various messaging middleware platforms. In particular, MAMA adds market data semantics to messaging platforms that would otherwise be too limited for use as a market data distribution middleware. Features that MAMA adds to any messaging middleware are:

- Subscription management (initial values, throttling).
- Entitlements/permissioning.
- Data quality.

The goal of MAMA is to maximize application portability. Once an application has been ported to MAMA, it should never have to be ported to another market data messaging API again. Many firms have invested time in building and maintaining their own abstraction APIs - and they should be commended for that. Wombat hopes that even those firms will see the benefit in migrating to MAMA and thereby reducing costs further and, as more third party firms migrate applications to MAMA (and MAMDA, see below), benefit even more from this compatibility.

A higher level market data API is also available: the Middleware Agnostic Market Data API (MAMDA). While MAMA provides a field-based abstraction to market data, MAMDA provides smart, specialized data types to deal with specific market data events, such as trades, quotes, order books, etc. MAMDA is particularly suitable for applications such as program trading and tick capture systems, where context is important. MAMA is more suited to applications that don't care about the meaning of fields, such as a simple, field-based market data display application.

Chapter 2

MAMA C++ API Namespace Index

2.1 MAMA C++ API Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

MAMA C++ API Hierarchical Index

3.1 MAMA C++ API Class Hierarchy

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

Chapter 4

MAMA C++ API Class Index

4.1 MAMA C++ API Class List

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

Chapter 5

MAMA C++ API File Index

5.1 MAMA C++ API File List

Here is a list of all files with brief descriptions:

Chapter 6

MAMA C++ API Namespace Documentation

6.1 std Namespace Reference

6.2 Wombat Namespace Reference

Classes

- class MamaBasicSubscription

  The `MamaBasicSubscription` *interface represents a subscription to a topic with no market data semantics.*

- class MamaBasicSubscriptionCallback

  The message callback interface for basic subscriptions.

- class MamaBasicWildCardSubscription

  The `MamaBasicWildCardSubscription` *interface represents a subscription to a topic with no market data semantics.*

- class MamaBasicWildCardSubscriptionCallback

  The message callback interface for basic subscriptions.

- class MamaBridgeCallback

  Bridge callback.

- class MamaLogFileCallback

  Subclass this to receive log notifications.

- class MamaStartCallback

  Callback object passed to Mama::startBackground().

- class Mama

  The `Mama` *class provides methods global initialization and manipulating global options.*

- class MamaDateTime

  A date/time representation with additional hints for precision, advanced output formatting and support for time zone conversion (using the MamaTimeZone type).

- class MamaDictionary

  The `MamaDictionary` *class maps field identifiers (FIDs) to human readable strings.*

- class MamaDictionaryCallback

  The `WombatDictionaryCallback` *receives notification regarding the population of the data dictionary.*

- class MamaDispatcher

  The dispatcher dispatches events from a queue until it is destroyed or MamaQueue->stopDispatch () is called.

- class MamaDQPublisher
- class MamaPublishTopic
- class MamaDQPublisherManager
- class MamaDQPublisherManagerCallback
- class MamaFieldDescriptor

  The `MamaFieldDescriptor` *class describes a field within a* `MamaDictionary`.

- class MamaFtMemberCallback
- class MamaFtMember
- class MamaMulticastFtMember
- class MamaBridgeFtMember
- class MamaInbox

  Used for sending messages requesting a direct reply.

- class MamaInboxCallback

  The MamaInboxCallback gets invoked when a message arrives in an inbox or when inbox related errors arise.

- class MamaIo

  A repeating IO.

- class MamaIoCallback

  Subclass this to receive IO notifications.

- class MamaLogFile

  The `MamaLogFile` *class provides a single interface for the configuration and control of* `Mama` *logging activity.*

- class MamaMsgIterator

  Iterator class for mamaMsg.

- class MamaMsg

  MAMA message representation.

- class MamaMsgField

  MamaMsg field representation.

- class MamaMsgFieldIterator

Callback class for iterating over all fields in a message.

- class MamaMsgQual

  The MamaMsgQual class is a wrapper/utility class which provides useful interrogation, comparison and manipulation facilities for the Mama Message Qualifier data field (wMsgQual) which is a "bit-map" used to convey duplicate, delayed and out-of-sequence information about messages.

- class MamaPrice

  MamaPrice is a special data type for representing floating point numbers that often require special formatting for display purposes, such as prices.

- class MamaPublisher

  The publisher class publishes messages to basic or market data subscriptions depending on the underlying transport.

- class MamaQueue

  Queue allows applications to dispatch events in order with multiple threads using a single MamaDispatcher for each queue.

- class MamaQueueEnqueueCallback

  Callback interface for the MamaQueue::setEnqueueCallback () method.

- class MamaQueueEventCallback

  Definition of the callback method for when a user added event fires.

- class MamaQueueGroup

  A simple class for allocating subscriptions amongst multiple queues in a round robin.

- class MamaQueueMonitorCallback

  Receive callbacks when certain conditions for the MamaQueue are met.

- class MamaReservedFields
- class MamaSendCompleteCallback

  Callback interface for use with the MamaPublisher.sendWithThrottle() and MamaPublisher.sendFromInboxWithThrottle() methods.

- class MamaSubscriptionIteratorCallback
- class MamaSource

  A MAMA source maintains information about a data source, including the quality of the data coming from that source.

- class MamaSourceDerivative

A MamaSourceDerivative provides a reference to a common MamaSource object but can have attributes (such as the quality state) overridden.

- class MamaSourceGroup

  A MAMA source group monitors a set of MAMA sources that presumably provide a duplicate set of data.

- class MamaSourceGroupManager

  A MAMA source group manager monitors a set of MAMA source groups.

- class MamaSourceManager

  A MAMA source manager maintains information about a set of data sources, including the quality of the data coming from those sources.

- class MamaSourceStateChangeCallback

  Applications can register with MamaSourceGroup to receive state change notifications when the state of sources within the group has changed.

- class MamaStat
- class MamaStatsCollector
- class MamaStatus
- class MamaSubscription

  The `MamaSubscription` *interface represents a subscription to a topic.*

- class MamaSubscriptionCallback

  The message callback interface.

- class MamaSymbolList

  MamaSymbolList manages a list of MAMA symbols and related attributes.

- class MamaSymbolListFile

  MamaSymbolListFile is a file based symbol list with the ability to detect external changes to the file.

- class MamaSymbolListIteratorHandler

  The MamaSymbolListIteratorHandler interface.

- class MamaSymbolListMembershipHandler

  The MamaSymbolListMembershipHandler interface.

- class MamaSymbolListResortHandler

  The MamaSymbolListResortHandler interface.

- class MamaSymbolListMember

  The MamaSymbolListMember class represents the information about the symbol list member.

- class MamaSymbolMap

  The MamaSymbolMap class provides the ability for MAMA to do client side symbology mapping.

- class MamaSymbolMapFile

  MamaSymbolMapFile is a concrete implementation of a symbology map.

- class MamaSymbolSource

  MamaSymbolSource defines an interface which all SymbolSources should implement in order to provide a mechanism by which objects implementing the "MamaSymbolSourceCallback::onSymbol" can be registered with the source such that they can be notified of new symbols as they arrive.

- class MamaSymbolSourceCallback

  The MamaSymbolSourceCallback interface.

- class MamaSymbolStoreSaveCallback

  The MamaSymbolStoreSaveCallback interface.

- class MamaTimer

  A repeating timer.

- class MamaTimerCallback

  Subclass this to receive timer notifications.

- class MamaTimeZone

  A time zone representation to make conversion of timestamps to and from particular time zones more convenient.

- class MamaTransportTopicEventCallback

  TransportTopicEvent callback.

- class MamaTransportCallback

  Transport callback.

- class MamaTransport

  The MamaTransport class provides market data functionality.

- class MamaTransportMap

Chapter 7

MAMA C++ API Class Documentation

7.1 Wombat::Mama Class Reference

The Mama class provides methods global initialization and manipulating global options.

`#include <mamacpp.h>`

Static Public Member Functions

- static mamaBridge loadBridge (const char *middleware)

  Load the bridge specified by middleware string.

- static mamaBridge loadBridge (const char *middleware, const char *path)

  Load the bridge specified by middleware string using the path specified by the user.

- static const char * getVersion (mamaBridge bridgeImpl)

  Returns the version of the mama binary.

- static void open ()

  Initialize MAMA.

- static void open (const char *path, const char *filename)

  Initialize MAMA.

- static void setProperty (const char *name, const char *value)

Set a specific property for the API.

- static const char * getProperty (const char *name)

  Retrieve a specific property from the API.

- static void close ()

  Close MAMA and free all associated resource.

- static void start (mamaBridge bridgeImpl)

  Start processing messages on the internal queue.

- static void startBackground (mamaBridge bridgeImpl, MamaStartCallback *callback)

  Start processing MAMA internal events in the background.

- static void stop (mamaBridge bridgeImpl)

  Stop dispatching on the default event queue for the specified bridge.

- static void stopAll (void)

  Stop dispatching on the default event queue for all bridges.

- static void enableLogging (MamaLogLevel level, FILE *logFile)

  Enable logging and direct the output to the specified stream.

- static void logToFile (const char *file, MamaLogLevel level)

  Enable logging to the specified file.

- static void disableLogging (void)

  Disable logging.

- static void setLogLevel (MamaLogLevel level)

  Set the logging level.

- static MamaLogLevel getLogLevel (void)

  Get the logging level.

- static void setLogSize (unsigned long size)

  Set the maxmum size of the log file (bytes) Default max size is 500 Mb.

- static void setNumLogFiles (int numFiles)

  Set the number of rolled logfiles to keep before overwriting.

- static void setLogFilePolicy (mamaLogFilePolicy policy)

Set the policy regarding how to handle files when Max file size is reached.

- static void setAppendToLogFile (bool append)

  Set the mode when opening an existing log file.

- static bool loggingToFile (void)

  Get the status of loggingToFile Returns true if logging to a file, false if not.

- static void setLogSizeCb (MamaLogFileCallback *callback)

  Set a callback for when the max log size is reached.

- static void setApplicationName (const char *applicationName)

  Set the mama application name This should be called before Mama.open().

- static void setApplicationClassName (const char *className)

  Set the mama application class This should be called before Mama.open().

- static MamaQueue * getDefaultEventQueue (mamaBridge bridgeImpl)

  Get a pointer to the internal default MAMA event queue.

- template<typename T> static void deleteObject (T *object)

  Allow the MAMA API free memory for any objects which have been allocated by the API but responsibility for deleting has been handed to the application code.

- static void setBridgeCallback (mamaBridge bridge, MamaBridgeCallback *callback)

  Set a MamaBridgeMessageCallback to be invoked whenever information messages are logged at the bridge level.

- static void addStatsCollector (MamaStatsCollector *statsCollector)

  It adds the newly created statsCollector to the statsGenerator list.

- static void removeStatsCollector (MamaStatsCollector *statsCollector)

  It removes the statsCollector from the statsGenerator list.

7.1.1 Detailed Description

The `Mama` class provides methods global initialization and manipulating global options.

7.1.2 Member Function Documentation

7.1.2.1 static mamaBridge Wombat::Mama::loadBridge (const char * *middleware*) [static]

Load the bridge specified by middleware string.

If the bridge has already been loaded then the existing bridge instance will be returned.

Parameters:

impl The bridge object

middleware The middleware string. Can be "wmw", "lbm" or "tibrv".

Returns:

mama_status Whether the call was successful or not.

7.1.2.2 static mamaBridge Wombat::Mama::loadBridge (const char * *middleware*, const char * *path*) [static]

Load the bridge specified by middleware string using the path specified by the user.

If the bridge has already been loaded then the existing bridge instance will be returned.

Parameters:

impl The bridge object

middleware The middleware string. Can be "wmw", "lbm" or "tibrv".

path The path to the bridge library

Returns:

mama_status Whether the call was successful or not.

7.1.2.3 static const char* Wombat::Mama::getVersion (mamaBridge *bridgeImpl*) [static]

Returns the version of the mama binary.

The version of the underlying transport is also returned in parens after the mama version.

7.1.2.4 static void Wombat::Mama::open () [static]

Initialize MAMA.

MAMA employs a reference count to track multiple calls to Mama::open() and Mama::close(). The count is incremented every time Mama::open() is called and decremented when Mama::close() is called. The resources are not actually released until the count reaches zero.

If entitlements are enabled for the library, the available entitlement server names are read from the entitlement.servers property in the mama.properties file located in the $WOMBAT_PATH directory.

This function is thread safe.

7.1.2.5 static void Wombat::Mama::open (const char * *path*, const char * *filename*) [static]

Initialize MAMA.

Allows users of the API to override the default behaviour of Mama.open() where a file mama.properties is required to be located in the directory specified by $WOMBAT_-PATH.

The properties file must have the same structure as a standard mama.properties file.

If null is passed as the path the API will look for the properties file on the $WOMBAT_-PATH.

If null is passed as the filename the API will look for the default filename of mama.properties.

Parameters:

← ***path*** Fully qualified path to the directory containing the properties file

← ***filename*** The name of the file containing MAMA properties.

7.1.2.6 static void Wombat::Mama::setProperty (const char * *name*, const char * *value*) [static]

Set a specific property for the API.

If the property being set has already been given a value from a properties file that value will be replaced.

See the example mama.properties provided with the distribution for examples of property formatting. The properties set via this function should be formatted in the same manner as those specified in mama.properties.

The strings passed to the function are copied.

Parameters:

name The name of the property

value The property value

7.1.2.7 static const char* Wombat::Mama::getProperty (const char * *name*) [static]

Retrieve a specific property from the API.

If the property has not been set, a NULL value will be returned.

Parameters:

name The name of the property to retrieve.

Returns:

the value of the property or NULL if unset.

7.1.2.8 static void Wombat::Mama::close () [static]

Close MAMA and free all associated resource.

MAMA employs a reference count to track multiple calls to Mama::open() and Mama::close(). The count is incremented every time Mama::open() is called and decremented when Mama::close() is called. The resources are not actually released until the count reaches zero.

This function is thread safe.

7.1.2.9 static void Wombat::Mama::start (mamaBridge *bridgeImpl*) [static]

Start processing messages on the internal queue.

This starts Mama's internal throttle, refresh logic, and other internal Mama processes as well as dispatching messages from the internal queue.

Mama::start( ) blocks until an invocation of Mama::stop() occurs.

MAMA employs a reference count to track multiple calls to Mama::start() and Mama::stop(). The count is incremented every time Mama::start() is called and decremented when Mama::stop() is called. The first Mama::start() call does not unblock until the count reaches zero.

This function is thread safe.

Parameters:

← ***bridgeImpl*** The bridge specific structure.

7.1.2.10 static void Wombat::Mama::startBackground (mamaBridge *bridgeImpl*, MamaStartCallback * *callback*) [static]

Start processing MAMA internal events in the background.

This method invokes Mama::start () in a separate thread.

Parameters:

← ***bridgeImpl*** The middleware-specific bridge structure

← ***callback*** The callback for asynchronous status.

7.1.2.11 static void Wombat::Mama::stop (mamaBridge *bridgeImpl*) [static]

Stop dispatching on the default event queue for the specified bridge.

MAMA employs a reference count to track multiple calls to Mama::start() and Mama::stop(). The count is incremented every time Mama::start() is called and decremented when Mama::stop() is called. The first Mama::start() call does not unblock until the count reaches zero.

This function is thread safe.

Parameters:

← ***bridgeImpl*** The bridge specific structure.

7.1.2.12 static void Wombat::Mama::stopAll (void) [static]

Stop dispatching on the default event queue for all bridges.

7.1.2.13 static void Wombat::Mama::enableLogging (MamaLogLevel *level*, FILE * *logFile*) [static]

Enable logging and direct the output to the specified stream.

Parameters:

level The level

logFile the log file.

7.1.2.14 static void Wombat::Mama::logToFile (const char * *file*, MamaLogLevel *level*) [static]

Enable logging to the specified file.

Parameters:

file the log filename

level The level

7.1.2.15 static void Wombat::Mama::disableLogging (void) [static]

Disable logging.

7.1.2.16 static void Wombat::Mama::setLogLevel (MamaLogLevel *level*) [static]

Set the logging level.

Parameters:

level The level

7.1.2.17 static MamaLogLevel Wombat::Mama::getLogLevel (void) [static]

Get the logging level.

Returns:

the logging level

7.1.2.18 static void Wombat::Mama::setLogSize (unsigned long *size*) [static]

Set the maxmum size of the log file (bytes) Default max size is 500 Mb.

Parameters:

size the max size of file (bytes)

7.1.2.19 static void Wombat::Mama::setNumLogFiles (int *numFiles*) [static]

Set the number of rolled logfiles to keep before overwriting.

Default is 10

Parameters:

numFiles the max number of logfiles

7.1.2.20 static void Wombat::Mama::setLogFilePolicy (mamaLogFilePolicy *policy*) [static]

Set the policy regarding how to handle files when Max file size is reached.

Default is LOGFILE_UNBOUNDED - uses a single logfile unlimited in size. Other policies are: LOGFILE_ROLL - keeps N logfiles specified with setNumLogFiles(N). LOGFILE_OVERWRITE - uses a single logfile limited in size. LOGFILE_USER - if user has registered a callback it will be calledt. Otherwise the file will roll or get overwritten depending onthe value specified with setNumLogFiles(N).

Parameters:

policy the policy to use when max size is reached

7.1.2.21 static void Wombat::Mama::setAppendToLogFile (bool *append*) [static]

Set the mode when opening an existing log file.

setAppendToLogFile(true) will add data to the end of an existing file. Default is false which will overwrite any existing data in the file.

Parameters:

append boolean flag to set append mode on or off

7.1.2.22 static bool Wombat::Mama::loggingToFile (void) [static]

Get the status of loggingToFile Returns true if logging to a file, false if not.

Returns:

the status of loggingToFile

7.1.2.23 static void Wombat::Mama::setLogSizeCb (MamaLogFileCallback * *callback*) [static]

Set a callback for when the max log size is reached.

This will only be called if the policy has been set to LOGFILE_USER.

Parameters:

LogSizeCallback function pointer for the callback

7.1.2.24 static void Wombat::Mama::setApplicationName (const char * *applicationName*) [static]

Set the mama application name This should be called before Mama.open().

Parameters:

applicationName

7.1.2.25 static void Wombat::Mama::setApplicationClassName (const char * *className*) [static]

Set the mama application class This should be called before Mama.open().

Parameters:

className

7.1.2.26 static MamaQueue* Wombat::Mama::getDefaultEventQueue (mamaBridge *bridgeImpl*) [static]

Get a pointer to the internal default MAMA event queue.

Parameters:

bridgeImpl The middleware specific bridge structure.

Returns:

A pointer to the internal MAMA default event queue.

7.1.2.27 template<typename T> static void Wombat::Mama::deleteObject (T * *object*) [static]

Allow the MAMA API free memory for any objects which have been allocated by the API but responsibility for deleting has been handed to the application code.

This enables users of the API to provide alternate memory management implementations which may result in difficulties when deleting objects allocated internally by the MAMA API.

E.g. Detaching the MamaMsg in a subscription callback.

Currently supported types:

MamaMsg

7.1.2.28 static void Wombat::Mama::setBridgeCallback (mamaBridge *bridge*, MamaBridgeCallback * *callback*) [static]

Set a MamaBridgeMessageCallback to be invoked whenever information messages are logged at the bridge level.

Information messages vary depenging on the underlying middleware. Currently only supported for LBM.

7.1.2.29 static void Wombat::Mama::addStatsCollector (MamaStatsCollector * *statsCollector*) [static]

It adds the newly created statsCollector to the statsGenerator list.

Parameters:

statsCollector

7.1.2.30 static void Wombat::Mama::removeStatsCollector (MamaStatsCollector * *statsCollector*) [static]

It removes the statsCollector from the statsGenerator list.

Parameters:

statsCollector

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

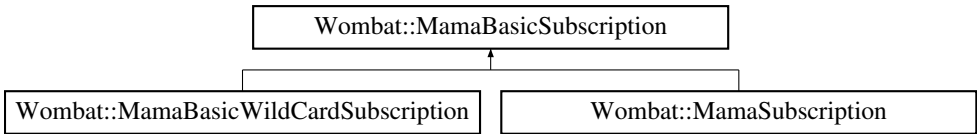
- mamacpp.h

7.2 Wombat::MamaBasicSubscription Class Reference

The MamaBasicSubscription interface represents a subscription to a topic with no market data semantics.

`#include <MamaBasicSubscription.h>`

Inheritance diagram for Wombat::MamaBasicSubscription::



Public Member Functions

- virtual ~MamaBasicSubscription ()

  The destructor will call `destroy()` if the subscription has not already been destroyed.

- MamaBasicSubscription (void)

  Constructor.

- virtual void createBasic (MamaTransport *transport, MamaQueue *queue, MamaBasicSubscriptionCallback *callback, const char *topic, void *closure=NULL)

  Create a basic subscription without market data semantics.

- virtual void destroy ()

  Destroy the subscription.

- virtual void destroyEx ()

  This function will destroy the subscription and can be called from any thread.

- virtual bool isActive (void) const

  Return whether the subscription is active.

- virtual const char * getTopic (void) const

  Return the symbol for this subscription.

- virtual MamaTransport * getTransport (void) const

Return the `MamaTransport` *for this subscription.*

- virtual MamaBasicSubscriptionCallback * getBasicCallback (void) const

  Return the `MamaSubscriptionCallback` *for this subscription.*

- virtual MamaQueue * getQueue (void) const

  Return the `MamaQueue` *for the subscription.*

- virtual void setClosure (void *closure)

  Set the closure for the subscription.

- virtual void * getClosure (void) const

  Return the closure for the subscription.

- virtual void setDebugLevel (MamaLogLevel level)

  Set the debug level for this subscription.

- virtual MamaLogLevel getDebugLevel () const

  Return the debug level for this subscription.

- virtual bool checkDebugLevel (MamaLogLevel level) const

  Return whether the debug level for this subscription equals or exceeds some level.

- virtual mamaSubscriptionState getState (void)

  This function will return the current state of the subscription, this function should be used in preference to the mamaSubscription_isActive or mamaSubscription_isValid functions.

Protected Attributes

- void * mClosure
- MamaQueue * mQueue
- mamaSubscription mSubscription
- MamaTransport * mTransport

7.2.1 Detailed Description

The `MamaBasicSubscription` interface represents a subscription to a topic with no market data semantics.

See also:

Mama

Author:

Michael Schonberg copyright 2003 Wombat Consulting Inc.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 virtual Wombat::MamaBasicSubscription::~MamaBasicSubscription () [virtual]

The destructor will call `destroy()` if the subscription has not already been destroyed.

7.2.2.2 Wombat::MamaBasicSubscription::MamaBasicSubscription (void)

Constructor.

Call createBasic() to activate the subscription.

7.2.3 Member Function Documentation

7.2.3.1 virtual void Wombat::MamaBasicSubscription::createBasic (MamaTransport * *transport*, MamaQueue * *queue*, MamaBasicSubscriptionCallback * *callback*, const char * *topic*, void * *closure* = NULL) [virtual]

Create a basic subscription without market data semantics.

Parameters:

transport The transport to use. Must be a basic transport.

queue The queue.

callback The mamaMsgCallbacks structure containing the three callback methods.

topic The topic.

closure The caller supplied closure.

7.2.3.2 virtual void Wombat::MamaBasicSubscription::destroy () [virtual]

Destroy the subscription.

Destroys the underlying subscription. The subscription can be recreated via a subsequent call to create()

Reimplemented in Wombat::MamaSubscription.

7.2.3.3 virtual void Wombat::MamaBasicSubscription::destroyEx ()

`[virtual]`

This function will destroy the subscription and can be called from any thread.

Note that the subscription will not be fully destroyed until the onDestroy callback is received from the MamaBasicSubscriptionCallback interface. To destroy from the dispatching thread the destroy function should be used in preference.

Reimplemented in Wombat::MamaSubscription.

7.2.3.4 virtual bool Wombat::MamaBasicSubscription::isActive (void) const

`[virtual]`

Return whether the subscription is active.

Returns:

whether the subscription is active.

7.2.3.5 virtual const char* Wombat::MamaBasicSubscription::getTopic (void) const

`[virtual]`

Return the symbol for this subscription.

Returns:

The topic.

7.2.3.6 virtual MamaTransport* Wombat::MamaBasicSubscription::getTransport (void) const

`[virtual]`

Return the `MamaTransport` for this subscription.

Returns:

the transport.

7.2.3.7 virtual MamaBasicSubscriptionCallback* Wombat::MamaBasicSubscription::getBasicCallback (void) const [virtual]

Return the MamaSubscriptionCallback for this subscription.

Returns:

the callback.

7.2.3.8 virtual MamaQueue* Wombat::MamaBasicSubscription::getQueue (void) const [virtual]

Return the MamaQueue for the subscription.

Returns:

the queue.

7.2.3.9 virtual void Wombat::MamaBasicSubscription::setClosure (void * *closure*) [virtual]

Set the closure for the subscription.

7.2.3.10 virtual void* Wombat::MamaBasicSubscription::getClosure (void) const [virtual]

Return the closure for the subscription.

Returns:

the closure.

7.2.3.11 virtual void Wombat::MamaBasicSubscription::setDebugLevel (MamaLogLevel *level*) [virtual]

Set the debug level for this subscription.

Parameters:

level The new debug level.

7.2.3.12 virtual MamaLogLevel Wombat::MamaBasicSubscription::getDebugLevel () const [virtual]

Return the debug level for this subscription.

Returns:

the debug level for this subscription.

7.2.3.13 virtual bool Wombat::MamaBasicSubscription::checkDebugLevel (MamaLogLevel *level*) const [virtual]

Return whether the debug level for this subscription equals or exceeds some level.

Parameters:

level The debug level to check.

Returns:

whether the level equals or exceeds the set level for this subscription.

7.2.3.14 virtual mamaSubscriptionState Wombat::MamaBasicSubscription::getState (void) [virtual]

This function will return the current state of the subscription, this function should be used in preference to the mamaSubscription_isActive or mamaSubscription_isValid functions.

This function is thread-safe.

Returns:

mama_status this can be one of the mamaSubscriptionState enumeration values.

7.2.4 Member Data Documentation

7.2.4.1 void* Wombat::MamaBasicSubscription::mClosure [protected]

7.2.4.2 MamaQueue* Wombat::MamaBasicSubscription::mQueue [protected]

7.2.4.3 mamaSubscription Wombat::MamaBasicSubscription::mSubscription [protected]

7.2.4.4 MamaTransport* Wombat::MamaBasicSubscription::mTransport [protected]

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

- MamaBasicSubscription.h

7.3 Wombat::MamaBasicSubscriptionCallback Class Reference

The message callback interface for basic subscriptions.

`#include <MamaBasicSubscriptionCallback.h>`

Public Member Functions

- virtual ~MamaBasicSubscriptionCallback ()
- virtual void onCreate (MamaBasicSubscription *subscription)=0

  Method invoked when subscription creation is complete, and before any calls to `onMsg`.

- virtual void onError (MamaBasicSubscription *subscription, const MamaStatus &status, const char *topic)=0

  Invoked if an error occurs during prior to subscription creation or if the subscription receives a message for an unentitled topic.

- virtual void onMsg (MamaBasicSubscription *subscription, MamaMsg &msg)=0

  Invoked when a message arrives.

- virtual void onDestroy (MamaBasicSubscription *subscription, void *closure)

  This method is invoked when a subscription has been completely destroyed, the client can have confidence that no further events will be placed on the queue for this subscription.

7.3.1 Detailed Description

The message callback interface for basic subscriptions.

Callers provide an object implementing this interface on creating a `MamaSubscription`.

See also:

MamaSubscription

Author:

mls

7.3.2 Constructor & Destructor Documentation

7.3.2.1 virtual Wombat::MamaBasicSubscriptionCallback::~MamaBasicSubscriptionCallback () [virtual]

```
42 {};
```

7.3.3 Member Function Documentation

7.3.3.1 virtual void Wombat::MamaBasicSubscriptionCallback::onCreate (MamaBasicSubscription * *subscription*) [pure virtual]

Method invoked when subscription creation is complete, and before any calls to `onMsg`.

Since subscriptions are created asynchronous by throttle, this callback provides the subscription instance after the throttle processes the creation request.

Parameters:

subscription The subscription.

7.3.3.2 virtual void Wombat::MamaBasicSubscriptionCallback::onError (MamaBasicSubscription * *subscription*, const MamaStatus & *status*, const char * *topic*) [pure virtual]

Invoked if an error occurs during prior to subscription creation or if the subscription receives a message for an unentitled topic.

If the status is `MamaMsgStatus.NOT_ENTITTLED` the topic parameter is the specific unentitled topic. If the subscription topic contains wildcards, the subscription may still receive messages for other entitled topics.

Parameters:

subscription The subscription.

status The wombat error code.

topic The topic for NOT_ENTITLED

7.3.3.3 virtual void Wombat::MamaBasicSubscriptionCallback::onMsg (MamaBasicSubscription * *subscription*, MamaMsg & *msg*) [pure virtual]

Invoked when a message arrives.

Parameters:

subscription the MamaSubscription.

msg The TibrvMsg.

7.3.3.4 virtual void Wombat::MamaBasicSubscriptionCallback::onDestroy (MamaBasicSubscription * *subscription*, void * *closure*) [virtual]

This method is invoked when a subscription has been completely destroyed, the client can have confidence that no further events will be placed on the queue for this subscription.

Parameters:

← ***subscription*** The The Mama Basic Subscription.

← ***closure*** The closure passed to the create function.

```
97      {
98      };
```

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

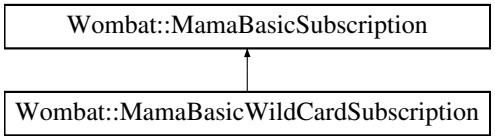
- MamaBasicSubscriptionCallback.h

7.4 Wombat::MamaBasicWildCardSubscription Class Reference

The MamaBasicWildCardSubscription interface represents a subscription to a topic with no market data semantics.

`#include <MamaBasicWildCardSubscription.h>`

Inheritance diagram for Wombat::MamaBasicWildCardSubscription::



Public Member Functions

- virtual ~MamaBasicWildCardSubscription ()

  The destructor will call `destroy()` if the subscription has not already been destroyed.

- MamaBasicWildCardSubscription (void)

  Constructor.

- virtual void create (MamaTransport *transport, MamaQueue *queue, MamaBasicWildCardSubscriptionCallback *callback, const char *source, const char *topic, void *closure=NULL)

  Create a basic wild card subscription without market data semantics.

- virtual const char * getSymbol (void) const

  Return the symbol for this subscription.

- virtual const char * getSource (void) const

  Return the source for this subscription.

- virtual void * getTopicClosure () const

  Get the closure for the specific wildcard topic.

- virtual void setTopicClosure (void *closure)

  Set the topic closure for the current message's topic.

- virtual void muteCurrentTopic (void)

For "transport" subscriptions (WMW only) stop processing messages for the current topic.

- virtual MamaBasicWildCardSubscriptionCallback * getBasicWildCardCallback (void) const

  Return the `MamaSubscriptionCallback` *for this subscription.*

7.4.1 Detailed Description

The `MamaBasicWildCardSubscription` interface represents a subscription to a topic with no market data semantics.

See also:

Mama

Author:

Michael Schonberg copyright 2003 Wombat Consulting Inc.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 virtual Wombat::MamaBasicWildCardSubscription::~MamaBasicWildCardSubscription () `[virtual]`

The destructor will call `destroy()` if the subscription has not already been destroyed.

7.4.2.2 Wombat::MamaBasicWildCardSubscription::MamaBasicWildCardSubscription (void)

Constructor.

Call createBasicWildCard() to activate the subscription.

7.4.3 Member Function Documentation

7.4.3.1 virtual void Wombat::MamaBasicWildCardSubscription::create (MamaTransport * *transport*, MamaQueue * *queue*, MamaBasicWildCardSubscriptionCallback * *callback*, const char * *source*, const char * *topic*, void * *closure* = NULL) `[virtual]`

Create a basic wild card subscription without market data semantics.

If the topic is NULL for WMW, this method creates a "transport" subscription that subscribes to all the topics from publishers with the the "publish_name" property equal to the source value.

Parameters:

transport The transport to use. Must be a basic transport.

queue The queue.

callback The mamaMsgCallbacks structure containing the three callback methods.

topic The topic.

closure The caller supplied closure.

7.4.3.2 virtual const char* Wombat::MamaBasicWildCardSubscription::getSymbol (void) const [virtual]

Return the symbol for this subscription.

Returns:

The topic.

7.4.3.3 virtual const char* Wombat::MamaBasicWildCardSubscription::getSource (void) const [virtual]

Return the source for this subscription.

Returns:

The topic.

7.4.3.4 virtual void* Wombat::MamaBasicWildCardSubscription::getTopicClosure () const [virtual]

Get the closure for the specific wildcard topic.

This method may only be called from the onMsg callback.

@ return The closure specified by setTopicClosure() or NULL if no topic closure set.

7.4.3.5 virtual void Wombat::MamaBasicWildCardSubscription::setTopicClosure (void * *closure*) [virtual]

Set the topic closure for the current message's topic.

This method can only be invoked from the onMsg() callback.

closure The closure.

7.4.3.6 virtual void Wombat::MamaBasicWildCardSubscription::muteCurrentTopic (void) [virtual]

For "transport" subscriptions (WMW only) stop processing messages for the current topic.

7.4.3.7 virtual MamaBasicWildCardSubscriptionCallback* Wombat::MamaBasicWildCardSubscription::getBasicWildCardCallback (void) const [virtual]

Return the MamaSubscriptionCallback for this subscription.

Returns:

the callback.

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

- MamaBasicWildCardSubscription.h

7.5 Wombat::MamaBasicWildCardSubscriptionCallback Class Reference

The message callback interface for basic subscriptions.

`#include <MamaBasicWildCardSubscriptionCallback.h>`

Public Member Functions

- virtual ~MamaBasicWildCardSubscriptionCallback ()
- virtual void onCreate (MamaBasicWildCardSubscription *subscription)=0

  Method invoked when subscription creation is complete, and before any calls to `onMsg`.

- virtual void onError (MamaBasicWildCardSubscription *subscription, const MamaStatus &status, const char *topic)=0

  Invoked if an error occurs during prior to subscription creation or if the subscription receives a message for an unentitled topic.

- virtual void onDestroy (MamaBasicWildCardSubscription *subscription, void *closure)

  This method is invoked when a subscription has been completely destroyed, the client can have confidence that no further events will be placed on the queue for this subscription.

- virtual void onMsg (MamaBasicWildCardSubscription *subscription, MamaMsg &msg, const char *topic)=0

  Invoked when a message arrives.

7.5.1 Detailed Description

The message callback interface for basic subscriptions.

Callers provide an object implementing this interface on creating a `MamaSubscription`.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 virtual Wombat::MamaBasicWildCardSubscriptionCallback::~MamaBasicWildCardSubscriptionCallback () [virtual]

```
38 {};
```

7.5.3 Member Function Documentation

7.5.3.1 virtual void Wombat::MamaBasicWildCardSubscriptionCallback::onCreate (MamaBasicWildCardSubscription * *subscription*) [pure virtual]

Method invoked when subscription creation is complete, and before any calls to `onMsg`.

Since subscriptions are created asynchronous by throttle, this callback provides the subscription instance after the throttle processes the creation request.

Parameters:

subscription The subscription.

7.5.3.2 virtual void Wombat::MamaBasicWildCardSubscriptionCallback::onError (MamaBasicWildCardSubscription * *subscription*, const MamaStatus & *status*, const char * *topic*) [pure virtual]

Invoked if an error occurs during prior to subscription creation or if the subscription receives a message for an unentitled topic.

If the status is `MamaMsgStatus.NOT_ENTITTLED` the topic parameter is the specific unentitled topic. If the subscription topic contains wildcards, the subscription may still receive messages for other entitled topics.

Parameters:

subscription The subscription.

status The wombat error code.

topic The topic for NOT_ENTITLED

7.5.3.3 virtual void Wombat::MamaBasicWildCardSubscriptionCallback::onDestroy (MamaBasicWildCardSubscription * *subscription*, void * *closure*) [virtual]

This method is invoked when a subscription has been completely destroyed, the client can have confidence that no further events will be placed on the queue for this subscription.

Parameters:

← ***subscription*** The The Mama Basic Wildcard Subscription.

← ***closure*** The closure passed to the create function.

```
80      {
81      };
```

7.5.3.4 virtual void Wombat::MamaBasicWildCardSubscriptionCallback::onMsg (MamaBasicWildCardSubscription * *subscription*, MamaMsg & *msg*, const char * *topic*) [pure virtual]

Invoked when a message arrives.

Parameters:

subscription the MamaSubscription.

msg The TibrvMsg.

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

- MamaBasicWildCardSubscriptionCallback.h

7.6 Wombat::MamaBridgeCallback Class Reference

Bridge callback.

`#include <MamaBridgeCallback.h>`

Public Member Functions

- virtual ~MamaBridgeCallback ()
- virtual void onBridgeInfo (mamaBridge bridgeImpl, const char *message)

  Invoked when an info message is generated at the bridge level.

7.6.1 Detailed Description

Bridge callback.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 virtual Wombat::MamaBridgeCallback::~MamaBridgeCallback () [virtual]

```
37 {};
```

7.6.3 Member Function Documentation

7.6.3.1 virtual void Wombat::MamaBridgeCallback::onBridgeInfo (mamaBridge *bridgeImpl*, const char * *message*) [virtual]

Invoked when an info message is generated at the bridge level.

Currently only available in LBM.

Parameters:

bridge The bridge which the message relates to.

message The message.

```
48     {
49         return;
50     }
```

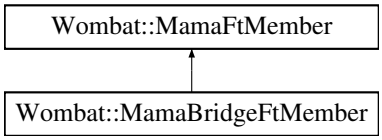
The documentation for this class was generated from the following file:

- MamaBridgeCallback.h

7.7 Wombat::MamaBridgeFtMember Class Reference

`#include <MamaFt.h>`

Inheritance diagram for Wombat::MamaBridgeFtMember::



Public Member Functions

- void setup (MamaQueue *queue, MamaFtMemberCallback *handler, MamaTransport *transport, const char *groupName, mama_u32_t weight, mama_f64_t heartbeatInterval, mama_f64_t timeoutInterval, void *closure=NULL)

7.7.1 Member Function Documentation

7.7.1.1 void Wombat::MamaBridgeFtMember::setup (MamaQueue * *queue*, MamaFtMemberCallback * *handler*, MamaTransport * *transport*, const char * *groupName*, mama_u32_t *weight*, mama_f64_t *heartbeatInterval*, mama_f64_t *timeoutInterval*, void * *closure* = NULL) [virtual]

Implements Wombat::MamaFtMember.

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

- MamaFt.h

7.8 Wombat::MamaDateTime Class Reference

A date/time representation with additional hints for precision, advanced output formatting and support for time zone conversion (using the MamaTimeZone type).

`#include <MamaDateTime.h>`

Public Member Functions

- MamaDateTime ()
- MamaDateTime (const MamaDateTime &copy)
- MamaDateTime (const char *str, const MamaTimeZone *tz=NULL)

  Constructor taking a string argument.

- ~MamaDateTime ()
- MamaDateTime & operator= (const MamaDateTime &rhs)
- bool operator== (const MamaDateTime &rhs) const
- bool operator!= (const MamaDateTime &rhs) const
- bool operator< (const MamaDateTime &rhs) const
- bool operator> (const MamaDateTime &rhs) const
- int compare (const MamaDateTime &rhs) const
- bool empty () const
- void setEpochTime (mama_u32_t secondsSinceEpoch, mama_u32_t microseconds, mamaDateTimePrecision precision=MAMA_DATE_TIME_PREC_UNKNOWN)
- void setEpochTimeF64 (double secondsSinceEpoch)
- void setEpochTimeMilliseconds (mama_u64_t millisecondsSinceEpoch)
- void setEpochTimeMicroseconds (mama_u64_t microsecondsSinceEpoch)
- void setWithHints (mama_u32_t secondsSinceEpoch, mama_u32_t microseconds, mamaDateTimePrecision precision=MAMA_DATE_TIME_PREC_UNKNOWN, mamaDateTimeHints hints=0)
- void setPrecision (mamaDateTimePrecision precision)
- void setFromString (const char *str, const MamaTimeZone *tz=NULL)
- void setFromString (const char *str, mama_size_t strLen, const MamaTimeZone *tz=NULL)
- void setToNow ()
- void setToMidnightToday (const MamaTimeZone *tz=NULL)
- void set (mama_u32_t year, mama_u32_t month, mama_u32_t day, mama_u32_t hour, mama_u32_t minute, mama_u32_t second, mama_u32_t microsecond, mamaDateTimePrecision precision=MAMA_DATE_TIME_PREC_UNKNOWN, const MamaTimeZone *tz=NULL)

  Set the entire date and time for the MamaDateTime.

- void setTime (mama_u32_t hour, mama_u32_t minute, mama_u32_t second, mama_u32_t microsecond, mamaDateTimePrecision precision=MAMA_DATE_TIME_PREC_UNKNOWN, const MamaTimeZone *tz=NULL)

  Set the time-of-day portion of the MamaDateTime.

- void setDate (mama_u32_t year, mama_u32_t month, mama_u32_t day)

  Set the date portion of the MamaDateTime.

- void copyTime (const MamaDateTime &copy)

  Copy the time-of-day portion of the MamaDateTime.

- void copyDate (const MamaDateTime &copy)

  Copy the date portion of the MamaDateTime.

- void clear ()

  Clear the entire MamaDateTime.

- void clearTime ()

  Clear the time-of-day portion of the MamaDateTime.

- void clearDate ()

  Clear the date portion of the MamaDateTime.

- void addSeconds (mama_f64_t seconds)
- void addSeconds (mama_i32_t seconds)
- void addMicroseconds (mama_i64_t microSeconds)
- mama_u64_t getEpochTimeMicroseconds () const

  Get the date and time as microseconds since the Epoch, (using the UTC timezone).

- mama_u64_t getEpochTimeMicroseconds (const MamaTimeZone &tz) const

  Get the date and time as microseconds since the Epoch in the supplied time zone.

- mama_u64_t getEpochTimeMilliseconds () const
- mama_u64_t getEpochTimeMilliseconds (const MamaTimeZone &tz) const
- mama_f64_t getEpochTimeSeconds () const
- mama_f64_t getEpochTimeSeconds (const MamaTimeZone &tz) const
- mama_f64_t getEpochTimeSecondsWithCheck () const

  Get the date and time as seconds since the Epoch, (using the UTC timezone).

- void getAsString (char *result, mama_size_t maxLen) const
- void getTimeAsString (char *result, mama_size_t maxLen) const
- void getDateAsString (char *result, mama_size_t maxLen) const
- const char * getAsString () const

Return a string representation of the date/time.

- const char * getTimeAsString () const
- const char * getDateAsString () const
- void getAsFormattedString (char *result, mama_size_t maxLen, const char *format) const
- void getAsFormattedString (char *result, mama_size_t maxLen, const char *format, const MamaTimeZone &tz) const
- void getAsStructTimeVal (struct timeval &result) const
- void getAsStructTimeVal (struct timeval &result, const MamaTimeZone &tz) const
- void getAsStructTm (struct tm &result) const
- void getAsStructTm (struct tm &result, const MamaTimeZone &tz) const
- mama_u32_t getYear () const
- mama_u32_t getMonth () const
- mama_u32_t getDay () const
- mama_u32_t getHour () const
- mama_u32_t getMinute () const
- mama_u32_t getSecond () const
- mama_u32_t getMicrosecond () const
- mamaDayOfWeek getDayOfWeek () const
- bool hasTime () const

  Return whether the object has a time component.

- bool hasDate () const

  Return whether the object has a date component.

- mamaDateTime getCValue ()
- const mamaDateTime getCValue () const

7.8.1 Detailed Description

A date/time representation with additional hints for precision, advanced output formatting and support for time zone conversion (using the MamaTimeZone type).

Hints include:

- Whether the time stamp contains a date part, a time part, or both.
- The level of accuracy (if known) of the time part (e.g., minutes, seconds, milliseconds, etc.).

The output format strings are similar to that available for the strftime() function with the addition of %; which adds optional (non-zero) fractional second to the string, and %: which adds fractional seconds including trailing zeros, but does not include the ".". The following table provides examples of output.

Actual Time	Output of "%T%;"	Output of "%T%.:"
01:23:45 and 678 millisecs	01:23:45.678	01:23:45.678
01:23:45 and 0 millisecs	01:23:45	01:23:45.000

7.8.2 Constructor & Destructor Documentation

7.8.2.1 Wombat::MamaDateTime::MamaDateTime ()

7.8.2.2 Wombat::MamaDateTime::MamaDateTime (const MamaDateTime & *copy*)

7.8.2.3 Wombat::MamaDateTime::MamaDateTime (const char * *str*, const MamaTimeZone * *tz* = NULL) [explicit]

Constructor taking a string argument.

This is the same as calling setFromString() immediately after construction.

7.8.2.4 Wombat::MamaDateTime::~MamaDateTime ()

7.8.3 Member Function Documentation

7.8.3.1 MamaDateTime& Wombat::MamaDateTime::operator= (const MamaDateTime & *rhs*)

7.8.3.2 bool Wombat::MamaDateTime::operator== (const MamaDateTime & *rhs*) const

7.8.3.3 bool Wombat::MamaDateTime::operator!= (const MamaDateTime & *rhs*) const

```
85     { return ! operator== (rhs); }
```

7.8.3.4 bool Wombat::MamaDateTime::operator< (const MamaDateTime & *rhs*) const

7.8.3.5 bool Wombat::MamaDateTime::operator> (const MamaDateTime & *rhs*) const

7.8.3.6 int Wombat::MamaDateTime::compare (const MamaDateTime & *rhs*) const

7.8.3.7 bool Wombat::MamaDateTime::empty () const

7.8.3.8 void Wombat::MamaDateTime::setEpochTime (mama_u32_t *secondsSinceEpoch*, mama_u32_t *microseconds*, mamaDateTimePrecision *precision* = MAMA_DATE_TIME_PREC_UNKNOWN)

7.8.3.9 void Wombat::MamaDateTime::setEpochTimeF64 (double *secondsSinceEpoch*)

7.8.3.10 void Wombat::MamaDateTime::setEpochTimeMilliseconds (mama_u64_t *millisecondsSinceEpoch*)

7.8.3.11 void Wombat::MamaDateTime::setEpochTimeMicroseconds (mama_u64_t *microsecondsSinceEpoch*)

7.8.3.12 void Wombat::MamaDateTime::setWithHints (mama_u32_t *secondsSinceEpoch*, mama_u32_t *microseconds*, mamaDateTimePrecision *precision* = MAMA_DATE_TIME_PREC_UNKNOWN, mamaDateTimeHints *hints* = 0)

7.8.3.13 void Wombat::MamaDateTime::setPrecision (mamaDateTimePrecision *precision*)

7.8.3.14 void Wombat::MamaDateTime::setFromString (const char * *str*, const MamaTimeZone * *tz* = NULL)

7.8.3.15 void Wombat::MamaDateTime::setFromString (const char * *str*, mama_size_t *strLen*, const MamaTimeZone * *tz* = NULL)

7.8.3.16 void Wombat::MamaDateTime::setToNow ()

7.8.3.17 void Wombat::MamaDateTime::setToMidnightToday (const MamaTimeZone * *tz* = NULL)

7.8.3.18 void Wombat::MamaDateTime::set (mama_u32_t *year*, mama_u32_t *month*, mama_u32_t *day*, mama_u32_t *hour*, mama_u32_t *minute*, mama_u32_t *second*, mama_u32_t *microsecond*, mamaDateTimePrecision *precision* = MAMA_DATE_TIME_PREC_UNKNOWN, const MamaTimeZone * *tz* = NULL)

Set the entire date and time for the MamaDateTime.

The year, month and day parameters must all be integers greater than zero.

Parameters:

year The year (must be 1970 or later).

month The month (1 - 12).

day The day (1 - 31).

hour The hour (0 - 23).

minute The minute (0 - 59).

second The second (0 - 59).

microsecond The second (0 - 999999).

precision An explicit precision, if known.

tz A timezone to shift from.

7.8.3.19 void Wombat::MamaDateTime::setTime (mama_u32_t *hour*, mama_u32_t *minute*, mama_u32_t *second*, mama_u32_t *microsecond*, mamaDateTimePrecision *precision* = MAMA_DATE_TIME_PREC_UNKNOWN, const MamaTimeZone * *tz* = NULL)

Set the time-of-day portion of the MamaDateTime.

The date portion is not affected.

Parameters:

hour The hour (0 - 23).

minute The minute (0 - 59).

second The second (0 - 59).

microsecond The second (0 - 999999).

precision An explicit precision, if known.

tz A timezone to shift from.

7.8.3.20 void Wombat::MamaDateTime::setDate (mama_u32_t *year*, mama_u32_t *month*, mama_u32_t *day*)

Set the date portion of the MamaDateTime.

The time-of-day portion is not affected. The year, month and day parameters must all be integers greater than zero.

Parameters:

year The year (must be 1970 or later).

month The month (1 - 12).

day The day (1 - 31).

7.8.3.21 void Wombat::MamaDateTime::copyTime (const MamaDateTime & *copy*)

Copy the time-of-day portion of the MamaDateTime.

The date portion is not affected.

Parameters:

copy The object to copy from

7.8.3.22 void Wombat::MamaDateTime::copyDate (const MamaDateTime & *copy*)

Copy the date portion of the MamaDateTime.

The time-of-day portion is not affected.

Parameters:

copy The object to copy from

7.8.3.23 void Wombat::MamaDateTime::clear ()

Clear the entire MamaDateTime.

7.8.3.24 void Wombat::MamaDateTime::clearTime ()

Clear the time-of-day portion of the MamaDateTime.

The date portion is not affected.

7.8.3.25 void Wombat::MamaDateTime::clearDate ()

Clear the date portion of the MamaDateTime.

The time-of-day portion is not affected.

7.8.3.26 void Wombat::MamaDateTime::addSeconds (mama_f64_t *seconds*)

7.8.3.27 void Wombat::MamaDateTime::addSeconds (mama_i32_t *seconds*)

7.8.3.28 void Wombat::MamaDateTime::addMicroseconds (mama_i64_t *microSeconds*)

7.8.3.29 mama_u64_t Wombat::MamaDateTime::getEpochTimeMicroseconds () const

Get the date and time as microseconds since the Epoch, (using the UTC timezone).

Returns:

The number of microseconds since the Epoch.

7.8.3.30 mama_u64_t Wombat::MamaDateTime::getEpochTimeMicroseconds (const MamaTimeZone & *tz*) const

Get the date and time as microseconds since the Epoch in the supplied time zone.

Parameters:

int] tz The timezone.

Returns:

The number of microseconds since the Epoch.

7.8.3.31 mama_u64_t Wombat::MamaDateTime::getEpochTimeMilliseconds () const

7.8.3.32 mama_u64_t Wombat::MamaDateTime::getEpochTimeMilliseconds (const MamaTimeZone & *tz*) const

7.8.3.33 mama_f64_t Wombat::MamaDateTime::getEpochTimeSeconds () const

7.8.3.34 mama_f64_t Wombat::MamaDateTime::getEpochTimeSeconds (const MamaTimeZone & *tz*) const

7.8.3.35 mama_f64_t Wombat::MamaDateTime::getEpochTimeSecondsWithCheck () const

Get the date and time as seconds since the Epoch, (using the UTC timezone).

If no date value is contained in the dateTime then it will be set to today's date and the calculation made.

Returns:

The number of seconds, (including partial seconds), since the Epoch.

7.8.3.36 void Wombat::MamaDateTime::getAsString (char * *result*, mama_size_t *maxLen*) const

7.8.3.37 void Wombat::MamaDateTime::getTimeAsString (char * *result*, mama_size_t *maxLen*) const

7.8.3.38 void Wombat::MamaDateTime::getDateAsString (char * *result*, mama_size_t *maxLen*) const

7.8.3.39 const char* Wombat::MamaDateTime::getAsString () const

Return a string representation of the date/time.

Note that the alternative getAsString() methods are more efficient because these method must allocate a temporary buffer (automatically destroyed upon object destruction).

7.8.3.40 const char* Wombat::MamaDateTime::getTimeAsString () const

7.8.3.41 const char* Wombat::MamaDateTime::getDateAsString () const

7.8.3.42 void Wombat::MamaDateTime::getAsFormattedString (char * *result*, mama_size_t *maxLen*, const char * *format*) const

7.8.3.43 void Wombat::MamaDateTime::getAsFormattedString (char * *result*, mama_size_t *maxLen*, const char * *format*, const MamaTimeZone & *tz*) const

7.8.3.44 void Wombat::MamaDateTime::getAsStructTimeVal (struct timeval & *result*) const

7.8.3.45 void Wombat::MamaDateTime::getAsStructTimeVal (struct timeval & *result*, const MamaTimeZone & *tz*) const

7.8.3.46 void Wombat::MamaDateTime::getAsStructTm (struct tm & *result*) const

7.8.3.47 void Wombat::MamaDateTime::getAsStructTm (struct tm & *result*, const MamaTimeZone & *tz*) const

7.8.3.48 mama_u32_t Wombat::MamaDateTime::getYear () const

7.8.3.49 mama_u32_t Wombat::MamaDateTime::getMonth () const

7.8.3.50 mama_u32_t Wombat::MamaDateTime::getDay () const

7.8.3.51 mama_u32_t Wombat::MamaDateTime::getHour () const

7.8.3.52 mama_u32_t Wombat::MamaDateTime::getMinute () const

7.8.3.53 mama_u32_t Wombat::MamaDateTime::getSecond () const

7.8.3.54 mama_u32_t Wombat::MamaDateTime::getMicrosecond () const

7.8.3.55 mamaDayOfWeek Wombat::MamaDateTime::getDayOfWeek () const

7.8.3.56 bool Wombat::MamaDateTime::hasTime () const

Return whether the object has a time component.

7.8.3.57 bool Wombat::MamaDateTime::hasDate () const

Return whether the object has a date component.

7.8.3.58 mamaDateTime Wombat::MamaDateTime::getCValue ()

7.8.3.59 const mamaDateTime Wombat::MamaDateTime::getCValue () const

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

- MamaDateTime.h

7.9 Wombat::MamaDictionary Class Reference

The MamaDictionary class maps field identifiers (FIDs) to human readable strings.

`#include <MamaDictionary.h>`

Public Member Functions

- virtual ~MamaDictionary ()
- MamaDictionary (void)
- virtual void create (MamaQueue *queue, MamaDictionaryCallback *callback, MamaSource *source, int retries=MAMA_DEFAULT_RETRIES, double timeout=MAMA_DEFAULT_TIMEOUT, void *closure=NULL)

  Create a dictionary subscription.

- virtual const char * getFeedName ()

  Return the dictionary source feed name.

- virtual const char * getFeedHost ()

  Return the dictionary source feed host.

- virtual MamaFieldDescriptor * getFieldByFid (mama_fid_t fid)

  Return the field with the specified field FID.

- virtual const MamaFieldDescriptor * getFieldByFid (mama_fid_t fid) const

  Return the field with the specified field FID.

- virtual MamaFieldDescriptor * getFieldByIndex (size_t index)

  Return the field with the corresponding zero based index.

- virtual const MamaFieldDescriptor * getFieldByIndex (size_t index) const

  Return the field with the corresponding zero based index.

- virtual MamaFieldDescriptor * getFieldByName (const char *name)

  Return the field with the specified name.

- virtual const MamaFieldDescriptor * getFieldByName (const char *name) const

  Return the field with the specified name.

- virtual mama_fid_t getMaxFid (void) const

  Return the highest field identifier.

- virtual size_t getSize (void) const

  Return the number of fields in the dictionary.

- virtual bool hasDuplicates (void) const

  Return true if there are multiple fields with the same name.

- virtual MamaDictionaryCallback * getCallback (void) const

  Return the callback.

- virtual void setCallback (MamaDictionaryCallback *callback)

  Set the callback to receive notifications when creation is complete or an error occurs.

- virtual mamaDictionary getDictC ()

  Return the underlying C mamaDictionary.

- virtual const mamaDictionary getDictC () const

  Return the underlying C mamaDictionary.

- virtual void * getClosure (void) const

  Return the closure for the dictionary.

- virtual MamaMsg * getDictionaryMessage () const

  Returns a MamaMsg representing the data dictionary.

- virtual void buildDictionaryFromMessage (MamaMsg &msg)

  Recreate a data dictionary from the MamaMsg supplied.

- virtual MamaFieldDescriptor * createFieldDescriptor (mama_fid_t fid, const char *name, mamaFieldType type)

  Add a new field descriptor to a dictionary.

- virtual void setMaxFid (mama_size_t maxFid)

  Tell the dictionary what the probable maximum fid in the data dictionary may be.

- virtual void writeToFile (const char *fileName)

  Write the data dictionary to a file.

- virtual void populateFromFile (const char *fileName)

  Populate a dictionary from the contents of a file.

Public Attributes

- DictionaryImpl * mPimpl

7.9.1 Detailed Description

The MamaDictionary class maps field identifiers (FIDs) to human readable strings.

Incoming MamaMsgs may contain FIDs but no field names. The dictionary allows applications to determine the name associated with a given FID. On some platforms, the inbound messages may have names, but not fids in which case the dictionary can map names to fids.

7.9.2 Constructor & Destructor Documentation

7.9.2.1 virtual Wombat::MamaDictionary::~MamaDictionary () [virtual]

7.9.2.2 Wombat::MamaDictionary::MamaDictionary (void)

7.9.3 Member Function Documentation

7.9.3.1 virtual void Wombat::MamaDictionary::create (MamaQueue * *queue*, MamaDictionaryCallback * *callback*, MamaSource * *source*, int *retries* = MAMA_DEFAULT_RETRIES, double *timeout* = MAMA_DEFAULT_TIMEOUT, void * *closure* = NULL) [virtual]

Create a dictionary subscription.

The caller supplied DictionaryCallback.onComplete will be invoked after the dictionary is fully constructed.

If there is an error creating the dictionary Mama invokes the onError callback, and the returned dictionary is not valid. In the event of a timeout, Mama invokes the onTimeout callback. This method uses the default timeout and retry values (SubscriptionBridge.DEFAULT_TIMEOUT and SubscriptionBridge.DEFAULT_RETRIES.

Parameters:

queue The mama queue.

callback The dictionary callback.

source The dictionary source. Depends upon feed handler configuration. See feed handler documentation for details

timeout The timeout in seconds.

retries The number of times to retry before failing.

closure The caller supplied closure.

7.9.3.2 virtual const char* Wombat::MamaDictionary::getFeedName () [virtual]

Return the dictionary source feed name.

Parameters:

dictionary The dictionary.

Returns:

the feed name

7.9.3.3 virtual const char* Wombat::MamaDictionary::getFeedHost () [virtual]

Return the dictionary source feed host.

Parameters:

dictionary The dictionary.

Returns:

the feed host

7.9.3.4 virtual MamaFieldDescriptor* Wombat::MamaDictionary::getFieldByFid (mama_fid_t *fid*) [virtual]

Return the field with the specified field FID.

This method is very efficient.

Parameters:

fid The field id.

Returns:

The field.

7.9.3.5 virtual const MamaFieldDescriptor* Wombat::MamaDictionary::getFieldByFid (mama_fid_t *fid*) const [virtual]

Return the field with the specified field FID.

This method is very efficient.

Parameters:

fid The field id.

Returns:

The field.

7.9.3.6 virtual MamaFieldDescriptor* Wombat::MamaDictionary::getFieldByIndex (size_t *index*) [virtual]

Return the field with the corresponding zero based index.

This method is O (N) with respect to the size of the dictionary.

Parameters:

index The index.

Returns:

The field.

7.9.3.7 virtual const MamaFieldDescriptor* Wombat::MamaDictionary::getFieldByIndex (size_t *index*) const [virtual]

Return the field with the corresponding zero based index.

This method is O (N) with respect to the size of the dictionary.

Parameters:

index The index.

Returns:

The field.

7.9.3.8 virtual MamaFieldDescriptor* Wombat::MamaDictionary::getFieldByName (const char * *name*) [virtual]

Return the field with the specified name.

If there is more than one field with the same name, the one with the lowest field id is returned.

Parameters:

name The name of the field.

Returns:

The field with the specified name or null if there is no such field.

7.9.3.9 virtual const MamaFieldDescriptor* Wombat::MamaDictionary::getFieldByName (const char * *name*) const [virtual]

Return the field with the specified name.

If there is more than one field with the same name, the one with the lowest field id is returned.

Parameters:

name The name of the field.

Returns:

The field with the specified name or null if there is no such field.

7.9.3.10 virtual mama_fid_t Wombat::MamaDictionary::getMaxFid (void) const [virtual]

Return the highest field identifier.

Returns:

The highest FID.

7.9.3.11 virtual size_t Wombat::MamaDictionary::getSize (void) const [virtual]

Return the number of fields in the dictionary.

Returns:

The number of entries in the dictionary.

7.9.3.12 virtual bool Wombat::MamaDictionary::hasDuplicates (void) const [virtual]

Return true if there are multiple fields with the same name.

Returns:

true if there are duplicates.

7.9.3.13 virtual MamaDictionaryCallback* Wombat::MamaDictionary::getCallback (void) const [virtual]

Return the callback.

See also:

MamaDictionaryCallback

Returns:

The callback

7.9.3.14 virtual void Wombat::MamaDictionary::setCallback (MamaDictionaryCallback * *callback*) [virtual]

Set the callback to receive notifications when creation is complete or an error occurs.

Parameters:

callback The callback.

7.9.3.15 virtual mamaDictionary Wombat::MamaDictionary::getDictC () [virtual]

Return the underlying C mamaDictionary.

7.9.3.16 virtual const mamaDictionary Wombat::MamaDictionary::getDictC () const [virtual]

Return the underlying C mamaDictionary.

7.9.3.17 virtual void* Wombat::MamaDictionary::getClosure (void) const [virtual]

Return the closure for the dictionary.

Returns:

the closure.

7.9.3.18 virtual MamaMsg* Wombat::MamaDictionary::getDictionaryMessage () const [virtual]

Returns a MamaMsg representing the data dictionary.

This message can be published or used to create a new MamaDictionary object. A new MamaMsg is created for each invocation of the method. It is the responsibility of the caller to delete the message when no longer needed.

Returns:

Pointer to a new MamaMsg for the dictionary.

7.9.3.19 virtual void Wombat::MamaDictionary::buildDictionaryFromMessage (MamaMsg & *msg*) [virtual]

Recreate a data dictionary from the MamaMsg supplied.

The MamaMsg is copied and can therefore be deleted after the method has returned.

Parameters:

msg Reference to a MamaMsg representing a data dictionary.

7.9.3.20 virtual MamaFieldDescriptor* Wombat::MamaDictionary::createFieldDescriptor (mama_fid_t *fid*, const char * *name*, mamaFieldType *type*) [virtual]

Add a new field descriptor to a dictionary.

New fields can be added to an existing dictionary obtained from the MAMA infrastructure. This function can also be used to manually build a data dictionary.

Parameters:

fid The fid for the new field descriptor.

name The name for the new field descriptor.

type The type for the new field descriptor.

7.9.3.21 virtual void Wombat::MamaDictionary::setMaxFid (mama_size_t *maxFid*) [virtual]

Tell the dictionary what the probable maximum fid in the data dictionary may be.

This is not necessary but will aid performance for manually creating a new dictionary or adding new fields to an existing dictionary.

Calling this function ensures that there is capacity in the dictionary for field descriptors with fids up to the maximum specified.

Fields with fids greater than specified can be added to the dictionary but this will incur the overhead of allocating more memory and copying dictionary elements.

Parameters:

maxFid The probable maximum fid being added to the dictionary.

7.9.3.22 virtual void Wombat::MamaDictionary::writeToFile (const char * *fileName*) [virtual]

Write the data dictionary to a file.

The dictionary will be written in the form: fid|fieldName|fieldType

Parameters:

fileName The name of the file to serialize the dictionary to. This can be a fully qualified name, relative or a file on the $WOMBAT_PATH

7.9.3.23 virtual void Wombat::MamaDictionary::populateFromFile (const char * *fileName*) [virtual]

Populate a dictionary from the contents of a file.

Can be used to add additional fields to an existing dictionary or to populate a new dictionary.

Parameters:

fileName The file from which to populate the dictionary. This can be a fully qualified name, relative or a file on the $WOMBAT_PATH

7.9.4 Member Data Documentation

7.9.4.1 DictionaryImpl* Wombat::MamaDictionary::mPimpl

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

- MamaDictionary.h

7.10 Wombat::MamaDictionaryCallback Class Reference

The `WombatDictionaryCallback` receives notification regarding the population of the data dictionary.

`#include <MamaDictionaryCallback.h>`

Public Member Functions

- virtual ~MamaDictionaryCallback (void)
- virtual void onTimeout (void)

  Called when a timeout occurs.

- virtual void onError (const char *message)

  Invoked when an error occurs.

- virtual void onComplete (void)

  Invoked when dictionary creation is complete.

7.10.1 Detailed Description

The `WombatDictionaryCallback` receives notification regarding the population of the data dictionary.

Clients implement the interface and pass it to `Mama.createDictionary-Subscription`

See also:

MamaDictionary
Mama::createDictionarySubscription

7.10.2 Constructor & Destructor Documentation

7.10.2.1 virtual Wombat::MamaDictionaryCallback::~MamaDictionary-Callback (void) [virtual]

`42 {}`

7.10.3 Member Function Documentation

7.10.3.1 virtual void Wombat::MamaDictionaryCallback::onTimeout (void) [virtual]

Called when a timeout occurs.

```
47 {}
```

7.10.3.2 virtual void Wombat::MamaDictionaryCallback::onError (const char * *message*) [virtual]

Invoked when an error occurs.

Parameters:

message The error message.

```
53 {}
```

7.10.3.3 virtual void Wombat::MamaDictionaryCallback::onComplete (void) [virtual]

Invoked when dictionary creation is complete.

```
58 {}
```

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

- MamaDictionaryCallback.h

7.11 Wombat::MamaDispatcher Class Reference

The dispatcher dispatches events from a queue until it is destroyed or MamaQueue->stopDispatch () is called.

`#include <MamaDispatcher.h>`

Public Member Functions

- ~MamaDispatcher (void)
- MamaDispatcher (void)
- void create (MamaQueue *queue)

  Create a mamaDispatcher.

- void destroy (void)

  Destroy the dispatcher;.

7.11.1 Detailed Description

The dispatcher dispatches events from a queue until it is destroyed or MamaQueue->stopDispatch () is called.

7.11.2 Constructor & Destructor Documentation

7.11.2.1 Wombat::MamaDispatcher::~MamaDispatcher (void)

7.11.2.2 Wombat::MamaDispatcher::MamaDispatcher (void)

7.11.3 Member Function Documentation

7.11.3.1 void Wombat::MamaDispatcher::create (MamaQueue * *queue*)

Create a mamaDispatcher.

The dispatcher spawns a thread to dispatch events from a queue. It will continue to dispatch events until it is destroyed or mamaQueue_stopDispatch is called.

Only a single dispatcher can be created for a given queue. Attempting to create multiple dispatchers for a queue will result in and error. Dispatching message from a single queue with multiple threads results in messages arriving out of order and sequence number gaps for market data subscriptions.

Parameters:

queue The MamaQueue.

7.11.3.2 void Wombat::MamaDispatcher::destroy (void)

Destroy the dispatcher;.

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

- MamaDispatcher.h

7.12 Wombat::MamaDQPublisher Class Reference

`#include <MamaDQPublisher.h>`

Public Member Functions

- virtual ~MamaDQPublisher (void)
- MamaDQPublisher (void)
- virtual void create (MamaTransport *transport, const char *topic)

  Create a MAMA DQ publisher for the corresponding transport.

- virtual void send (MamaMsg *msg)
- virtual void sendReply (const MamaMsg &request, MamaMsg *reply) const
- virtual void sendReply (mamaMsgReply replyHandle, MamaMsg *reply) const
- virtual void destroy (void)
- virtual void setStatus (mamaMsgStatus status)
- virtual void setSenderId (uint64_t id)
- virtual void setSeqNum (mama_seqnum_t num)
- virtual void * getCache (void)
- virtual void setCache (void *cache)

Protected Attributes

- MamaDQPublisherImpl * mImpl

Friends

- class MamaDQPublisherManagerImpl

7.12.1 Constructor & Destructor Documentation

7.12.1.1 virtual Wombat::MamaDQPublisher::~MamaDQPublisher (void) [virtual]

7.12.1.2 Wombat::MamaDQPublisher::MamaDQPublisher (void)

7.12.2 Member Function Documentation

7.12.2.1 virtual void Wombat::MamaDQPublisher::create (MamaTransport * *transport*, const char * *topic*) [virtual]

Create a MAMA DQ publisher for the corresponding transport.

Parameters:

transport The transport to use. Must be a basic transport.

topic for basic publishers. Symbol for market data topics.

7.12.2.2 virtual void Wombat::MamaDQPublisher::send (MamaMsg * *msg*) [virtual]

7.12.2.3 virtual void Wombat::MamaDQPublisher::sendReply (const MamaMsg & *request*, MamaMsg * *reply*) const [virtual]

7.12.2.4 virtual void Wombat::MamaDQPublisher::sendReply (mamaMsgReply *replyHandle*, MamaMsg * *reply*) const [virtual]

7.12.2.5 virtual void Wombat::MamaDQPublisher::destroy (void) [virtual]

7.12.2.6 virtual void Wombat::MamaDQPublisher::setStatus (mamaMsgStatus *status*) [virtual]

7.12.2.7 virtual void Wombat::MamaDQPublisher::setSenderId (uint64_t *id*) [virtual]

7.12.2.8 virtual void Wombat::MamaDQPublisher::setSeqNum (mama_seqnum_t *num*) [virtual]

7.12.2.9 virtual void* Wombat::MamaDQPublisher::getCache (void) [virtual]

7.12.2.10 virtual void Wombat::MamaDQPublisher::setCache (void * *cache*) [virtual]

7.12.3 Friends And Related Function Documentation

7.12.3.1 friend class MamaDQPublisherManagerImpl [friend]

7.12.4 Member Data Documentation

7.12.4.1 MamaDQPublisherImpl* Wombat::MamaDQPublisher::mImpl [protected]

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

- MamaDQPublisher.h

7.13 Wombat::MamaDQPublisherManager Class Reference

`#include <MamaDQPublisherManager.h>`

Public Member Functions

- virtual ~MamaDQPublisherManager (void)
- MamaDQPublisherManager (void)
- virtual void create (MamaTransport *transport, MamaQueue *queue, MamaDQPublisherManagerCallback *callback, const char *sourcename, const char *root="_MD")

  Create a MAMA publisher manager for the corresponding transport.

- virtual void addPublisher (const char *symbol, MamaDQPublisher *pub, void *cache)
- virtual MamaDQPublisher * removePublisher (const char *symbol)
- virtual MamaDQPublisher * createPublisher (const char *symbol, void *cache)
- virtual void destroyPublisher (const char *symbol)
- virtual void destroy (void)
- virtual void setStatus (mamaMsgStatus status)
- virtual void setSenderId (uint64_t id)
- virtual void setSeqNum (mama_seqnum_t num)
- virtual void sendSyncRequest (mama_u16_t nummsg, mama_f64_t delay, mama_f64_t duration)
- virtual void sendNoSubscribers (const char *symbol)

7.13.1 Constructor & Destructor Documentation

7.13.1.1 virtual Wombat::MamaDQPublisherManager::~MamaDQPublisherManager (void) [virtual]

7.13.1.2 Wombat::MamaDQPublisherManager::MamaDQPublisherManager (void)

7.13.2 Member Function Documentation

7.13.2.1 virtual void Wombat::MamaDQPublisherManager::create (MamaTransport * *transport*, MamaQueue * *queue*, MamaDQPublisherManagerCallback * *callback*, const char * *sourcename*, const char * *root* = "_MD") [virtual]

Create a MAMA publisher manager for the corresponding transport.

Parameters:

transport The transport to use. Must be a basic transport.

queue the quese subscription requests are received on.

callback subscription request callback.

sourcename The Sourcename for subscribers to send requests

root The root for market data publishers.

7.13.2.2 virtual void Wombat::MamaDQPublisherManager::addPublisher (const char * *symbol*, MamaDQPublisher * *pub*, void * *cache*) [virtual]

7.13.2.3 virtual MamaDQPublisher* Wombat::MamaDQPublisherManager::removePublisher (const char * *symbol*) [virtual]

7.13.2.4 virtual MamaDQPublisher* Wombat::MamaDQPublisherManager::createPublisher (const char * *symbol*, void * *cache*) [virtual]

7.13.2.5 virtual void Wombat::MamaDQPublisherManager::destroyPublisher (const char * *symbol*) [virtual]

7.13.2.6 virtual void Wombat::MamaDQPublisherManager::destroy (void) [virtual]

7.13.2.7 virtual void Wombat::MamaDQPublisherManager::setStatus (mamaMsgStatus *status*) [virtual]

7.13.2.8 virtual void Wombat::MamaDQPublisherManager::setSenderId (uint64_t *id*) [virtual]

7.13.2.9 virtual void Wombat::MamaDQPublisherManager::setSeqNum (mama_seqnum_t *num*) [virtual]

7.13.2.10 virtual void Wombat::MamaDQPublisherManager::sendSyncRequest (mama_u16_t *nummsg*, mama_f64_t *delay*, mama_f64_t *duration*) [virtual]

7.13.2.11 virtual void Wombat::MamaDQPublisherManager::sendNoSubscribers (const char * *symbol*) [virtual]

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

- MamaDQPublisherManager.h

7.14 Wombat::MamaDQPublisherManagerCallback Class Reference

`#include <MamaDQPublisherManagerCallback.h>`

Public Member Functions

- virtual ~MamaDQPublisherManagerCallback ()
- virtual void onCreate (MamaDQPublisherManager *publisherManager)=0
- virtual void onNewRequest (MamaDQPublisherManager *publisherManager, const char *symbol, short subType, short msgType, MamaMsg &msg)=0
- virtual void onRequest (MamaDQPublisherManager *publisherManager, const MamaPublishTopic &publishTopicInfo, short subType, short msgType, MamaMsg &msg)=0
- virtual void onRefresh (MamaDQPublisherManager *publisherManager, const MamaPublishTopic &publishTopicInfo, short subType, short msgType, MamaMsg &msg)=0
- virtual void onError (MamaDQPublisherManager *publisher, const MamaStatus &status, const char *errortxt, MamaMsg *msg)=0
- virtual void onMsg (MamaDQPublisherManager *publisher, MamaMsg &msg)

7.14.1 Constructor & Destructor Documentation

7.14.1.1 virtual Wombat::MamaDQPublisherManagerCallback::~MamaDQPublisherManagerCallback () [virtual]

```
40 {};
```

7.14.2 Member Function Documentation

7.14.2.1 virtual void Wombat::MamaDQPublisherManagerCallback::onCreate (MamaDQPublisherManager * *publisherManager*) [pure virtual]

7.14.2.2 virtual void Wombat::MamaDQPublisherManagerCallback::onNewRequest (MamaDQPublisherManager * *publisherManager*, const char * *symbol*, short *subType*, short *msgType*, MamaMsg & *msg*) [pure virtual]

7.14.2.3 virtual void Wombat::MamaDQPublisherManagerCallback::onRequest (MamaDQPublisherManager * *publisherManager*, const MamaPublishTopic & *publishTopicInfo*, short *subType*, short *msgType*, MamaMsg & *msg*) [pure virtual]

7.14.2.4 virtual void Wombat::MamaDQPublisherManagerCallback::onRefresh (MamaDQPublisherManager * *publisherManager*, const MamaPublishTopic & *publishTopicInfo*, short *subType*, short *msgType*, MamaMsg & *msg*) [pure virtual]

7.14.2.5 virtual void Wombat::MamaDQPublisherManagerCallback::onError (MamaDQPublisherManager * *publisher*, const MamaStatus & *status*, const char * *errortxt*, MamaMsg * *msg*) [pure virtual]

7.14.2.6 virtual void Wombat::MamaDQPublisherManagerCallback::onMsg (MamaDQPublisherManager * *publisher*, MamaMsg & *msg*) [virtual]

```
75                                          {};
```

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

- MamaDQPublisherManagerCallback.h

7.15 Wombat::MamaFieldDescriptor Class Reference

The MamaFieldDescriptor class describes a field within a MamaDictionary.

`#include <MamaFieldDescriptor.h>`

Public Member Functions

- virtual ~MamaFieldDescriptor ()
- MamaFieldDescriptor (mamaFieldDescriptor field)
- MamaFieldDescriptor (mama_fid_t fid, mamaFieldType type, const char *name)

  Create a new field descriptor based on supplied info.

- mama_fid_t getFid (void) const

  Return the field identifier.

- mamaFieldType getType (void) const

  Return the data type.

- const char * getName (void) const

  Return the human readable name of the field.

- const char * getTypeName (void) const

  Return a human readable string for mamaMsgType.

- void setClosure (void *closure)

  Associate user supplied data with the field descriptor.

- void * getClosure () const

  Return the user supplied data associated with the field descriptor.

- void setTrackModState (bool on)

  Track the modification state of the field.

- bool getTrackModState () const

  Track the modification state of the field?

- void setPubName (const char *pubName)

  Set the publish name for this field.

- const char * getPubName () const

  Get the publish name for this field.

Public Attributes

- FieldDescriptorImpl * mPimpl

7.15.1 Detailed Description

The MamaFieldDescriptor class describes a field within a MamaDictionary.

See also:

MamaDictionary
MamaMsg

7.15.2 Constructor & Destructor Documentation

7.15.2.1 virtual Wombat::MamaFieldDescriptor::~MamaFieldDescriptor () [virtual]

7.15.2.2 Wombat::MamaFieldDescriptor::MamaFieldDescriptor (mamaFieldDescriptor *field*)

7.15.2.3 Wombat::MamaFieldDescriptor::MamaFieldDescriptor (mama_fid_t *fid*, mamaFieldType *type*, const char * *name*)

Create a new field descriptor based on supplied info.

Parameters:

fid The field id.

type The field type.

name The field name.

7.15.3 Member Function Documentation

7.15.3.1 mama_fid_t Wombat::MamaFieldDescriptor::getFid (void) const

Return the field identifier.

Returns:

The fid.

7.15.3.2 mamaFieldType Wombat::MamaFieldDescriptor::getType (void) const

Return the data type.

Returns:

The type.

7.15.3.3 const char* Wombat::MamaFieldDescriptor::getName (void) const

Return the human readable name of the field.

Returns:

The name.

7.15.3.4 const char* Wombat::MamaFieldDescriptor::getTypeName (void) const

Return a human readable string for mamaMsgType.

7.15.3.5 void Wombat::MamaFieldDescriptor::setClosure (void * *closure*)

Associate user supplied data with the field descriptor.

Parameters:

closure The user supplied data to associate with the field descriptor.

7.15.3.6 void* Wombat::MamaFieldDescriptor::getClosure () const

Return the user supplied data associated with the field descriptor.

Returns:

The user supplied data associated with the field descriptor.

7.15.3.7 void Wombat::MamaFieldDescriptor::setTrackModState (bool *on*)

Track the modification state of the field.

7.15.3.8 bool Wombat::MamaFieldDescriptor::getTrackModState () const

Track the modification state of the field?

7.15.3.9 void Wombat::MamaFieldDescriptor::setPubName (const char * *pubName*)

Set the publish name for this field.

7.15.3.10 const char* Wombat::MamaFieldDescriptor::getPubName () const

Get the publish name for this field.

7.15.4 Member Data Documentation

7.15.4.1 FieldDescriptorImpl* Wombat::MamaFieldDescriptor::mPimpl

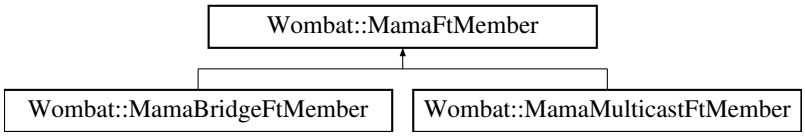
The documentation for this class was generated from the following file:

- MamaFieldDescriptor.h

7.16 Wombat::MamaFtMember Class Reference

`#include <MamaFt.h>`

Inheritance diagram for Wombat::MamaFtMember::



Public Member Functions

- MamaFtMember ()
- virtual ~MamaFtMember ()
- virtual void setup (MamaQueue *queue, MamaFtMemberCallback *handler, MamaTransport *transport, const char *groupName, mama_u32_t weight, mama_f64_t heartbeatInterval, mama_f64_t timeoutInterval, void *closure=NULL)=0
- void destroy ()
- void activate ()
- void deactivate ()
- bool isActive () const
- mamaFtState getState () const
- const char * getGroupName () const
- mama_u32_t getWeight () const
- mama_f64_t getHeartbeatInterval () const
- mama_f64_t getTimeoutInterval () const
- MamaFtMemberCallback * getCallback () const
- void * getClosure () const
- void setWeight (mama_u32_t weight)
- void setInstanceId (const char *id)
- mamaFtMember getCValue ()
- const mamaFtMember getCValue () const

Protected Attributes

- mamaFtMember mCValue
- MamaFtMemberCallback * mCallback
- void * mClosure

7.16.1 Constructor & Destructor Documentation

7.16.1.1 Wombat::MamaFtMember::MamaFtMember ()

7.16.1.2 virtual Wombat::MamaFtMember::~MamaFtMember ()

`[virtual]`

7.16.2 Member Function Documentation

7.16.2.1 virtual void Wombat::MamaFtMember::setup (MamaQueue * *queue*, MamaFtMemberCallback * *handler*, MamaTransport * *transport*, const char * *groupName*, mama_u32_t *weight*, mama_f64_t *heartbeatInterval*, mama_f64_t *timeoutInterval*, void * *closure* = NULL)

`[pure virtual]`

Implemented in Wombat::MamaMulticastFtMember, and Wombat::MamaBridgeFtMember.

7.16.2.2 void Wombat::MamaFtMember::destroy ()

7.16.2.3 void Wombat::MamaFtMember::activate ()

7.16.2.4 void Wombat::MamaFtMember::deactivate ()

7.16.2.5 bool Wombat::MamaFtMember::isActive () const

7.16.2.6 mamaFtState Wombat::MamaFtMember::getState () const

7.16.2.7 const char* Wombat::MamaFtMember::getGroupName () const

7.16.2.8 mama_u32_t Wombat::MamaFtMember::getWeight () const

7.16.2.9 mama_f64_t Wombat::MamaFtMember::getHeartbeatInterval () const

7.16.2.10 mama_f64_t Wombat::MamaFtMember::getTimeoutInterval () const

7.16.2.11 MamaFtMemberCallback* Wombat::MamaFtMember::getCallback () const

7.16.2.12 void* Wombat::MamaFtMember::getClosure () const

7.16.2.13 void Wombat::MamaFtMember::setWeight (mama_u32_t *weight*)

7.16.2.14 void Wombat::MamaFtMember::setInstanceId (const char * *id*)

7.16.2.15 mamaFtMember Wombat::MamaFtMember::getCValue ()

```
78          {
79              return mCValue;
80          }
```

7.16.2.16 const mamaFtMember Wombat::MamaFtMember::getCValue () const

```
82          {
83              return mCValue;
84          }
```

7.16.3 Member Data Documentation

7.16.3.1 mamaFtMember Wombat::MamaFtMember::mCValue [protected]

7.16.3.2 MamaFtMemberCallback* Wombat::MamaFtMember::mCallback [protected]

7.16.3.3 void* Wombat::MamaFtMember::mClosure [protected]

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

- MamaFt.h

7.17 Wombat::MamaFtMemberCallback Class Reference

`#include <MamaFt.h>`

Public Member Functions

- virtual ~MamaFtMemberCallback ()
- virtual void onFtStateChange (MamaFtMember *member, const char *groupName, mamaFtState state)=0

7.17.1 Constructor & Destructor Documentation

7.17.1.1 virtual Wombat::MamaFtMemberCallback::~MamaFtMemberCallback () [virtual]

```
37          {
38          };
```

7.17.2 Member Function Documentation

7.17.2.1 virtual void Wombat::MamaFtMemberCallback::onFtStateChange (MamaFtMember * *member*, const char * *groupName*, mamaFtState *state*) [pure virtual]

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

- MamaFt.h

7.18 Wombat::MamaInbox Class Reference

Used for sending messages requesting a direct reply.

`#include <MamaInbox.h>`

Public Member Functions

- virtual ~MamaInbox (void)
- MamaInbox (void)
- virtual void create (MamaTransport *transport, MamaQueue *queue, MamaInboxCallback *callback, void *closure=NULL)

  Create an inbox.

- virtual void destroy (void)
- virtual void * getClosure (void) const
- mamaInbox getCValue ()
- const mamaInbox getCValue () const

7.18.1 Detailed Description

Used for sending messages requesting a direct reply.

7.18.2 Constructor & Destructor Documentation

7.18.2.1 virtual Wombat::MamaInbox::~MamaInbox (void) [virtual]

7.18.2.2 Wombat::MamaInbox::MamaInbox (void)

7.18.3 Member Function Documentation

7.18.3.1 virtual void Wombat::MamaInbox::create (MamaTransport * *transport*, MamaQueue * *queue*, MamaInboxCallback * *callback*, void * *closure* = NULL) [virtual]

Create an inbox.

Parameters:

transport The transport for sending requests and receiving replies.

queue The queue.

callback The callback for receiving replies and errors.

closure The caller supplied closure.

7.18.3.2 virtual void Wombat::MamaInbox::destroy (void) [virtual]

7.18.3.3 virtual void* Wombat::MamaInbox::getClosure (void) const [virtual]

7.18.3.4 mamaInbox Wombat::MamaInbox::getCValue ()

7.18.3.5 const mamaInbox Wombat::MamaInbox::getCValue () const

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

- MamaInbox.h

7.19 Wombat::MamaInboxCallback Class Reference

The MamaInboxCallback gets invoked when a message arrives in an inbox or when inbox related errors arise.

`#include <MamaInboxCallback.h>`

Public Member Functions

- virtual ~MamaInboxCallback (void)
- virtual void onDestroy (MamaInbox *inbox, void *closure)

  This method is invoked when an inbox has been completely destroyed, the client can have confidence that no further events will be placed on the queue for this inbox.

- virtual void onMsg (MamaInbox *inbox, MamaMsg &msg)=0
- virtual void onError (MamaInbox *inbox, const MamaStatus &status)=0

7.19.1 Detailed Description

The MamaInboxCallback gets invoked when a message arrives in an inbox or when inbox related errors arise.

7.19.2 Constructor & Destructor Documentation

7.19.2.1 virtual Wombat::MamaInboxCallback::~MamaInboxCallback (void) [virtual]

```
37          {
38          };
```

7.19.3 Member Function Documentation

7.19.3.1 virtual void Wombat::MamaInboxCallback::onDestroy (MamaInbox * *inbox*, void * *closure*) [virtual]

This method is invoked when an inbox has been completely destroyed, the client can have confidence that no further events will be placed on the queue for this inbox.

Parameters:

← ***inbox*** The MamaInbox.

← ***closure*** The closure passed to the create function.

```
50          {
51          };
```

7.19.3.2 virtual void Wombat::MamaInboxCallback::onMsg (MamaInbox * *inbox*, MamaMsg & *msg*) [pure virtual]

7.19.3.3 virtual void Wombat::MamaInboxCallback::onError (MamaInbox * *inbox*, const MamaStatus & *status*) [pure virtual]

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

- MamaInboxCallback.h

7.20 Wombat::MamaIo Class Reference

A repeating IO.

`#include <MamaIo.h>`

Public Member Functions

- virtual ~MamaIo (void)
- MamaIo (void)
- virtual void create (MamaQueue *queue, MamaIoCallback *action, uint32_t descriptor, mamaIoType ioType, void *closure=NULL)

  Create an IO handler.

- virtual uint32_t getDescriptor (void) const
- virtual void destroy ()
- virtual void * getClosure (void) const

  Return the closure for the IO.

Public Attributes

- MamaIoImpl * mPimpl

7.20.1 Detailed Description

A repeating IO.

The callback will be repeatedly called at the specified interval until the IO is destroyed. See Mama::createMamaIo ().

The IO relies on the underlying middleware so its resolution is also dependent on the middleware.

7.20.2 Constructor & Destructor Documentation

7.20.2.1 virtual Wombat::MamaIo::~MamaIo (void) [virtual]

7.20.2.2 Wombat::MamaIo::MamaIo (void)

7.20.3 Member Function Documentation

7.20.3.1 virtual void Wombat::MamaIo::create (MamaQueue * *queue*, MamaIoCallback * *action*, uint32_t *descriptor*, mamaIoType *ioType*, void * *closure* = NULL) [virtual]

Create an IO handler.

Parameters:

queue The event queue for the io events. NULL specifies the Mama default queue.

action The callback to be invoked when an event occurs.

descriptor Wait for IO on this descriptor.

ioType Wait for occurrences of this type. See mama/io.h

closure The caller supplied closure.

7.20.3.2 virtual uint32_t Wombat::MamaIo::getDescriptor (void) const [virtual]

7.20.3.3 virtual void Wombat::MamaIo::destroy () [virtual]

7.20.3.4 virtual void* Wombat::MamaIo::getClosure (void) const [virtual]

Return the closure for the IO.

Returns:

the closure.

7.20.4 Member Data Documentation

7.20.4.1 MamaIoImpl* Wombat::MamaIo::mPimpl

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

- MamaIo.h

7.21 Wombat::MamaIoCallback Class Reference

Subclass this to receive IO notifications.

`#include <MamaIoCallback.h>`

Public Member Functions

- virtual ~MamaIoCallback (void)
- virtual void onIo (MamaIo *io, mamaIoType ioType)=0

7.21.1 Detailed Description

Subclass this to receive IO notifications.

7.21.2 Constructor & Destructor Documentation

7.21.2.1 virtual Wombat::MamaIoCallback::~MamaIoCallback (void) [virtual]

```
40         {
41         }
```

7.21.3 Member Function Documentation

7.21.3.1 virtual void Wombat::MamaIoCallback::onIo (MamaIo * *io*, mamaIoType *ioType*) [pure virtual]

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

- MamaIoCallback.h

7.22 Wombat::MamaLogFile Class Reference

The MamaLogFile class provides a single interface for the configuration and control of Mama logging activity.

`#include <MamaLogFile.h>`

Static Public Member Functions

- static void setMaxSize (unsigned long newMax)

  Set the Size of the log files.

- static void setNumBackups (unsigned int newNum)

  Set the number of log files to be maintained.

- static void setAppendMode (bool append)

  Set the open method for the logfile.

- static void enableLogging (const char *file, MamaLogLevel level)

  Enable logging.

- static void disableLogging (void)

  Disable logging.

- static bool loggingToFile (void)

  Return true if currently logging to file (via MamaLogFile).

- static void rollFiles ()

  Perform a log file rolling.

7.22.1 Detailed Description

The MamaLogFile class provides a single interface for the configuration and control of Mama logging activity.

The MamaLogFile class expands upon the existing logging capabilities of Mama providing the ability to set log file size and configure the number of log files maintained.

See also:

MamaLogFile

7.22.2 Member Function Documentation

7.22.2.1 static void Wombat::MamaLogFile::setMaxSize (unsigned long *newMax*) [static]

Set the Size of the log files.

Has no effect if log file is not successfully created via a call to MamaLogFile::enableLogging(...).

Parameters:

newMax log file size.

7.22.2.2 static void Wombat::MamaLogFile::setNumBackups (unsigned int *newNum*) [static]

Set the number of log files to be maintained.

Has no effect if log file is not successfully created via a call to MamaLogFile::enableLogging(...).

Parameters:

newNum number of log files to be maintained.

7.22.2.3 static void Wombat::MamaLogFile::setAppendMode (bool *append*) [static]

Set the open method for the logfile.

'true' will keep any existing data in the file 'false' will overwrite any existing data in the file

Parameters:

append append mode (on or off)

7.22.2.4 static void Wombat::MamaLogFile::enableLogging (const char * *file*, MamaLogLevel *level*) [static]

Enable logging.

Will set log level and attempt to open a file of the named provided. If a file could not be created or parameter file is NULL mama log output is re-directed to stderr.

Parameters:

file log file name

level mama log level

7.22.2.5 static void Wombat::MamaLogFile::disableLogging (void) [static]

Disable logging.

7.22.2.6 static bool Wombat::MamaLogFile::loggingToFile (void) [static]

Return true if currently logging to file (via MamaLogFile).

Return false if not logging to file (via MamaLogFile).

7.22.2.7 static void Wombat::MamaLogFile::rollFiles () [static]

Perform a log file rolling.

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

- MamaLogFile.h

7.23 Wombat::MamaLogFileCallback Class Reference

Subclass this to receive log notifications.

`#include <mamacpp.h>`

Public Member Functions

- virtual ~MamaLogFileCallback ()
- virtual void onLogSizeExceeded ()=0

7.23.1 Detailed Description

Subclass this to receive log notifications.

7.23.2 Constructor & Destructor Documentation

7.23.2.1 virtual Wombat::MamaLogFileCallback::~MamaLogFileCallback () [virtual]

```
120 {}
```

7.23.3 Member Function Documentation

7.23.3.1 virtual void Wombat::MamaLogFileCallback::onLogSizeExceeded () [pure virtual]

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

- mamacpp.h

7.24 Wombat::MamaMsg Class Reference

MAMA message representation.

`#include <MamaMsg.h>`

Public Member Functions

- ~MamaMsg ()
- MamaMsg (void)
- MamaMsg (const MamaMsg &mm)
- MamaMsg (WombatMsg &wm)
- void create (void)

  Create the actual underlying wire format message.

- void createForPayload (const char id)

  Create a mamaMsg.

- void createForPayloadBridge (mamaPayloadBridge payloadBridge)

  Create a mamaMsg.

- void createForWombatMsg (void)

  The underlying wire format for the message will be Wombat Message regardless of the middleware being used.

- void createFromBuffer (const void *buffer, size_t bufferLength)

  Create a MamaMsg from the provided byte buffer.

- void createForBridgeFromBuffer (const void *buffer, size_t bufferLength, mamaBridge bridge)

  Create a mamaMsg from the provided byte buffer using either the native format for the bridge (e.g.

- void copy (const MamaMsg &rhs)

  Copy the message from another, severing all links to the original message.

- MamaMsg * getTempCopy ()

  Get a temporary copy of the mamaMsg so to be able to modify the content.

- void applyMsg (const MamaMsg &msg)
- void clear (void)

  Clear the message.

- size_t getNumFields (void) const
  Returns the total number of fields in the message.
- size_t getByteSize (void) const
  Get the message size in bytes.
- bool getBoolean (const char *name, mama_fid_t fid) const
  Get a boolean field.
- bool getBoolean (const MamaFieldDescriptor *fieldDesc) const
  Get a boolean field.
- char getChar (const char *name, mama_fid_t fid) const
  Get a char field.
- char getChar (const MamaFieldDescriptor *fieldDesc) const
  Get a char field.
- mama_i8_t getI8 (const char *name, mama_fid_t fid) const
  Get an I8 field.
- mama_i8_t getI8 (const MamaFieldDescriptor *fieldDesc) const
  Get a I8 field.
- mama_u8_t getU8 (const char *name, mama_fid_t fid) const
  Get a U8 field.
- mama_u8_t getU8 (const MamaFieldDescriptor *fieldDesc) const
  Get a U8 field.
- mama_i16_t getI16 (const char *name, mama_fid_t fid) const
  Get an I16 field.
- mama_i16_t getI16 (const MamaFieldDescriptor *fieldDesc) const
  Get an I16 field.
- mama_u16_t getU16 (const char *name, mama_fid_t fid) const
  Get a U16 field.
- mama_u16_t getU16 (const MamaFieldDescriptor *fieldDesc) const
  Get an U16 field.

- mama_i32_t getI32 (const char *name, mama_fid_t fid) const
  Get an I32 field.
- mama_i32_t getI32 (const MamaFieldDescriptor *fieldDesc) const
  Get a I32 field.
- mama_u32_t getU32 (const char *name, mama_fid_t fid) const
  Get a U32 field.
- mama_u32_t getU32 (const MamaFieldDescriptor *fieldDesc) const
  Get a U32 field.
- mama_i64_t getI64 (const char *name, mama_fid_t fid) const
  Get a I64 field.
- mama_i64_t getI64 (const MamaFieldDescriptor *fieldDesc) const
  Get a I64 field.
- mama_u64_t getU64 (const char *name, mama_fid_t fid) const
  Get a U64 field.
- mama_u64_t getU64 (const MamaFieldDescriptor *fieldDesc) const
  Get a U64 field.
- mama_f32_t getF32 (const char *name, mama_fid_t fid) const
  Get a f32 field.
- mama_f32_t getF32 (const MamaFieldDescriptor *fieldDesc) const
  Get a f32 field.
- mama_f64_t getF64 (const char *name, mama_fid_t fid) const
  Get a f64 field.
- mama_f64_t getF64 (const MamaFieldDescriptor *fieldDesc) const
  Get a f64 field.
- const char * getString (const char *name, mama_fid_t fid) const
  Get a const char field.*
- const char * getString (const MamaFieldDescriptor *fieldDesc) const
  Get a const char field.*

- const void * getOpaque (const char *name, mama_fid_t fid, size_t &size) const

  Get an opaque field.

- const void * getOpaque (const MamaFieldDescriptor *fieldDesc, size_t &size) const

  Get a const char field.*

- void getDateTime (const char *name, mama_fid_t fid, MamaDateTime &result) const

  Get a MamaDateTime field.

- void getDateTime (const MamaFieldDescriptor *fieldDesc, MamaDateTime &result) const

  Get a MamaDateTime field.

- void getPrice (const char *name, mama_fid_t fid, MamaPrice &result) const

  Get a MamaPrice field.

- void getPrice (const MamaFieldDescriptor *fieldDesc, MamaPrice &result) const

  Get a MamaPrice field.

- const MamaMsg * getMsg (const char *name, mama_fid_t fid) const

  Get a submessage field.

- const MamaMsg * getMsg (const MamaFieldDescriptor *fieldDesc) const

  Get a submessage field.

- const char * getVectorChar (const char *name, mama_fid_t fid, size_t &result-Len) const

  Get a vector of characters.

- const char * getVectorChar (const MamaFieldDescriptor *fieldDesc, size_-t &resultLen) const

  Get a vector of characters.

- const mama_i8_t * getVectorI8 (const char *name, mama_fid_t fid, size_-t &resultLen) const

  Get a vector of signed 8-bit integers.

- const mama_i8_t * getVectorI8 (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of signed 8-bit integers.

- const mama_u8_t * getVectorU8 (const char *name, mama_fid_t fid, size_t &resultLen) const

  Get a vector of unsigned 8-bit integers.

- const mama_u8_t * getVectorU8 (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of unsigned 8-bit integers.

- const mama_i16_t * getVectorI16 (const char *name, mama_fid_t fid, size_t &resultLen) const

  Get a vector of signed 16-bit integers.

- const mama_i16_t * getVectorI16 (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of signed 16-bit integers.

- const mama_u16_t * getVectorU16 (const char *name, mama_fid_t fid, size_t &resultLen) const

  Get a vector of unsigned 16-bit integers.

- const mama_u16_t * getVectorU16 (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of unsigned 16-bit integers.

- const mama_i32_t * getVectorI32 (const char *name, mama_fid_t fid, size_t &resultLen) const

  Get a vector of signed 32-bit integers.

- const mama_i32_t * getVectorI32 (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of signed 32-bit integers.

- const mama_u32_t * getVectorU32 (const char *name, mama_fid_t fid, size_t &resultLen) const

  Get a vector of unsigned 32-bit integers.

- const mama_u32_t * getVectorU32 (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of unsigned 32-bit integers.

- const mama_i64_t * getVectorI64 (const char *name, mama_fid_t fid, size_t &resultLen) const

Get a vector of signed 64-bit integers.

- const mama_i64_t * getVectorI64 (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of signed 64-bit integers.

- const mama_u64_t * getVectorU64 (const char *name, mama_fid_t fid, size_t &resultLen) const

  Get a vector of unsigned 64-bit integers.

- const mama_u64_t * getVectorU64 (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of unsigned 64-bit integers.

- const mama_f32_t * getVectorF32 (const char *name, mama_fid_t fid, size_t &resultLen) const

  Get a vector of 32-bit floating point numbers.

- const mama_f32_t * getVectorF32 (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of 32-bit floating point numbers.

- const mama_f64_t * getVectorF64 (const char *name, mama_fid_t fid, size_t &resultLen) const

  Get a vector of 64-bit floating point numbers.

- const mama_f64_t * getVectorF64 (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of 64-bit floating point numbers.

- const MamaMsg ** getVectorMsg (const char *name, mama_fid_t fid, size_t &resultLen) const

  Get a vector of submessages field.

- const MamaMsg ** getVectorMsg (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

  Get a vector of submessages.

- const char ** getVectorString (const char *name, mama_fid_t fid, size_t &resultLen) const

  Get a vector of strings field.

- const char ** getVectorString (const MamaFieldDescriptor *fieldDesc, size_t &resultLen) const

Get a vector of submessages.

- bool tryBoolean (const char *name, mama_fid_t fid, bool &result) const
  Try to get a boolean field.
- bool tryBoolean (const MamaFieldDescriptor *field, bool &result) const
  Try to get a boolean field.
- bool tryChar (const char *name, mama_fid_t fid, char &result) const
  Try to get a char field.
- bool tryChar (const MamaFieldDescriptor *field, char &result) const
  Try to get a char field.
- bool tryI8 (const char *name, mama_fid_t fid, mama_i8_t &result) const
  Try to get an unsigned 8 bit integer field.
- bool tryI8 (const MamaFieldDescriptor *field, mama_i8_t &result) const
  Try to get an unsigned 8 bit field.
- bool tryU8 (const char *name, mama_fid_t fid, mama_u8_t &result) const
  Try to get an unsigned 8 bit integer field.
- bool tryU8 (const MamaFieldDescriptor *field, mama_u8_t &result) const
  Try to get an unsigned 8 bit field.
- bool tryI16 (const char *name, mama_fid_t fid, mama_i16_t &result) const
  Try to get a signed 16 bit integer field.
- bool tryI16 (const MamaFieldDescriptor *field, mama_i16_t &result) const
  Try to get a signed 16 bit field.
- bool tryU16 (const char *name, mama_fid_t fid, mama_u16_t &result) const
  Try to get an unsigned 16 bit integer field.
- bool tryU16 (const MamaFieldDescriptor *field, mama_u16_t &result) const
  Try to get an unsigned 16 bit field.
- bool tryI32 (const char *name, mama_fid_t fid, mama_i32_t &result) const
  Try to get a signed 32 bit integer field.
- bool tryI32 (const MamaFieldDescriptor *field, mama_i32_t &result) const

Try to get a signed 32 bit integer field.

- bool tryU32 (const char *name, mama_fid_t fid, mama_u32_t &result) const

  Try to get an unsigned 32 bit integer field.

- bool tryU32 (const MamaFieldDescriptor *field, mama_u32_t &result) const

  Try to get an unsigned 32 bit integer field.

- bool tryI64 (const char *name, mama_fid_t fid, mama_i64_t &result) const

  Try to get a signed 64 bit field.

- bool tryI64 (const MamaFieldDescriptor *field, mama_i64_t &result) const

  Try to get a signed 64 bit field.

- bool tryU64 (const char *name, mama_fid_t fid, mama_u64_t &result) const

  Try to get an unsigned 64 bit field.

- bool tryU64 (const MamaFieldDescriptor *field, mama_u64_t &result) const

  Try to get an unsigned 64 bit field.

- bool tryF32 (const char *name, mama_fid_t fid, mama_f32_t &result) const

  Try to get a f32 field.

- bool tryF32 (const MamaFieldDescriptor *field, mama_f32_t &result) const

  Try to get a f32 field.

- bool tryF64 (const char *name, mama_fid_t fid, mama_f64_t &result) const

  Try to get a f64 field.

- bool tryF64 (const MamaFieldDescriptor *field, mama_f64_t &result) const

  Try to get a f64 field.

- bool tryString (const char *name, mama_fid_t fid, const char *&result) const

  Try to get a string field.

- bool tryString (const MamaFieldDescriptor *field, const char *&result) const

  Try to get a string field.

- bool tryDateTime (const char *name, mama_fid_t fid, MamaDateTime &result) const

  Try to get a date/time field.

- bool tryDateTime (const MamaFieldDescriptor *field, MamaDateTime &result) const

  Try to get a date/time field.

- bool tryPrice (const char *name, mama_fid_t fid, MamaPrice &result) const

  Try to get a price field.

- bool tryPrice (const MamaFieldDescriptor *field, MamaPrice &result) const

  Try to get a price field.

- bool tryMsg (const char *name, mama_fid_t fid, const MamaMsg *&result) const

  Try to get a submessage field.

- bool tryMsg (const MamaFieldDescriptor *field, const MamaMsg *&result) const

  Try to get a submessage field.

- bool tryOpaque (const char *name, mama_fid_t fid, const void *&result, size_t &size) const

  Try to get a string field.

- bool tryOpaque (const MamaFieldDescriptor *field, const void *&result, size_t &size) const

  Try to get a string field.

- bool tryVectorChar (const char *name, mama_fid_t fid, const char *&result, size_t &resultLen) const

  Try to get a vector of characters.

- bool tryVectorChar (const MamaFieldDescriptor *fieldDesc, const char *&result, size_t &resultLen) const

  Try to get a vector of characters.

- bool tryVectorI8 (const char *name, mama_fid_t fid, const mama_i8_t *&result, size_t &resultLen) const

  Try to get a vector of signed 8-bit integers.

- bool tryVectorI8 (const MamaFieldDescriptor *fieldDesc, const mama_i8_t *&result, size_t &resultLen) const

  Try to get a vector of signed 8-bit integers.

- bool tryVectorU8 (const char *name, mama_fid_t fid, const mama_u8_t *&result, size_t &resultLen) const

  Try to get a vector of unsigned 8-bit integers.

- bool tryVectorU8 (const MamaFieldDescriptor *fieldDesc, const mama_u8_t *&result, size_t &resultLen) const

  Try to get a vector of unsigned 8-bit integers.

- bool tryVectorI16 (const char *name, mama_fid_t fid, const mama_i16_t *&result, size_t &resultLen) const

  Try to get a vector of signed 16-bit integers.

- bool tryVectorI16 (const MamaFieldDescriptor *fieldDesc, const mama_i16_t *&result, size_t &resultLen) const

  Try to get a vector of signed 16-bit integers.

- bool tryVectorU16 (const char *name, mama_fid_t fid, const mama_u16_t *&result, size_t &resultLen) const

  Try to get a vector of unsigned 16-bit integers.

- bool tryVectorU16 (const MamaFieldDescriptor *fieldDesc, const mama_u16_t *&result, size_t &resultLen) const

  Try to get a vector of unsigned 16-bit integers.

- bool tryVectorI32 (const char *name, mama_fid_t fid, const mama_i32_t *&result, size_t &resultLen) const

  Try to get a vector of signed 32-bit integers.

- bool tryVectorI32 (const MamaFieldDescriptor *fieldDesc, const mama_i32_t *&result, size_t &resultLen) const

  Try to get a vector of signed 32-bit integers.

- bool tryVectorU32 (const char *name, mama_fid_t fid, const mama_u32_t *&result, size_t &resultLen) const

  Try to get a vector of unsigned 32-bit integers.

- bool tryVectorU32 (const MamaFieldDescriptor *fieldDesc, const mama_u32_t *&result, size_t &resultLen) const

  Try to get a vector of unsigned 32-bit integers.

- bool tryVectorI64 (const char *name, mama_fid_t fid, const mama_i64_t *&result, size_t &resultLen) const

  Try to get a vector of signed 64-bit integers.

- bool tryVectorI64 (const MamaFieldDescriptor *fieldDesc, const mama_i64_t *&result, size_t &resultLen) const

  Try to get a vector of signed 64-bit integers.

- bool tryVectorU64 (const char *name, mama_fid_t fid, const mama_u64_t *&result, size_t &resultLen) const

  Try to get a vector of unsigned 64-bit integers.

- bool tryVectorU64 (const MamaFieldDescriptor *fieldDesc, const mama_u64_t *&result, size_t &resultLen) const

  Try to get a vector of unsigned 64-bit integers.

- bool tryVectorF32 (const char *name, mama_fid_t fid, const mama_f32_t *&result, size_t &resultLen) const

  Try to get a vector of 32-bit floating point numbers.

- bool tryVectorF32 (const MamaFieldDescriptor *fieldDesc, const mama_f32_t *&result, size_t &resultLen) const

  Try to get a vector of 32-bit floating point numbers.

- bool tryVectorF64 (const char *name, mama_fid_t fid, const mama_f64_t *&result, size_t &resultLen) const

  Try to get a vector of 64-bit floating point numbers.

- bool tryVectorF64 (const MamaFieldDescriptor *fieldDesc, const mama_f64_t *&result, size_t &resultLen) const

  Try to get a vector of 64-bit floating point numbers.

- bool tryVectorString (const char *name, mama_fid_t fid, const char **&result, size_t &resultLen) const

  Try to get a vector of strings.

- bool tryVectorString (const MamaFieldDescriptor *fieldDesc, const char **&result, size_t &resultLen) const

  Try to get a vector of strings.

- bool tryVectorMsg (const char *name, mama_fid_t fid, const MamaMsg **&result, size_t &resultLen) const

  Try to get a vector of submessages field.

- bool tryVectorMsg (const MamaFieldDescriptor *fieldDesc, const MamaMsg **&result, size_t &resultLen) const

Try to get a vector of submessages.

- void addBoolean (const char *name, mama_fid_t fid, bool value)
  Add a new boolean field.
- void addBoolean (const MamaFieldDescriptor *fieldDesc, bool value)
  Add a new boolean field.
- void addChar (const char *name, mama_fid_t fid, char value)
  Add a new char field.
- void addChar (const MamaFieldDescriptor *fieldDesc, char value)
  Add a new char field.
- void addI8 (const char *name, mama_fid_t fid, mama_i8_t value)
  Add a new I8 field.
- void addI8 (const MamaFieldDescriptor *fieldDesc, mama_i8_t value)
  Add a new I8 field.
- void addI16 (const char *name, mama_fid_t fid, mama_i16_t value)
  Add a new I16 field.
- void addI16 (const MamaFieldDescriptor *fieldDesc, mama_i16_t value)
  Add a new I16 field.
- void addI32 (const char *name, mama_fid_t fid, mama_i32_t value)
  Add a new I32 field.
- void addI32 (const MamaFieldDescriptor *fieldDesc, mama_i32_t value)
  Add a new I32 field.
- void addI64 (const char *name, mama_fid_t fid, mama_i64_t value)
  Add a new I64 field.
- void addI64 (const MamaFieldDescriptor *fieldDesc, mama_i64_t value)
  Add a new I64 field.
- void addU8 (const char *name, mama_fid_t fid, mama_u8_t value)
  Add a new byte (U8) const field.
- void addU8 (const MamaFieldDescriptor *fieldDesc, mama_u8_t value)

Add a new U8 field.

- void addU16 (const char *name, mama_fid_t fid, mama_u16_t value)

  Add a new short (U16) const field.

- void addU16 (const MamaFieldDescriptor *fieldDesc, mama_u16_t value)

  Add a new U16 field.

- void addU32 (const char *name, mama_fid_t fid, mama_u32_t value)

  Add a new int (U32) const field.

- void addU32 (const MamaFieldDescriptor *fieldDesc, mama_u32_t value)

  Add a new U32 field.

- void addU64 (const char *name, mama_fid_t fid, mama_u64_t value)

  Add a new int (U64) const field.

- void addU64 (const MamaFieldDescriptor *fieldDesc, mama_u64_t value)

  Add a new U64 field.

- void addF32 (const char *name, mama_fid_t fid, mama_f32_t value)

  Add a new F32 field.

- void addF32 (const MamaFieldDescriptor *fieldDesc, mama_f32_t value)

  Add a new F32 field.

- void addF64 (const char *name, mama_fid_t fid, mama_f64_t value)

  Add a new F64 field.

- void addF64 (const MamaFieldDescriptor *fieldDesc, mama_f64_t value)

  Add a new F64 field.

- void addString (const char *name, mama_fid_t fid, const char *value)

  Add a const char field.*

- void addString (const MamaFieldDescriptor *fieldDesc, const char *value)

  Add a new const char field.*

- void addDateTime (const char *name, mama_fid_t fid, const MamaDateTime &value)

  Add a date/time field.

- void addDateTime (const MamaFieldDescriptor *fieldDesc, const MamaDateTime &value)

  Add a new date/time field.

- void addPrice (const char *name, mama_fid_t fid, const MamaPrice &value)

  Add a price field.

- void addPrice (const MamaFieldDescriptor *fieldDesc, const MamaPrice &value)

  Add a new price field.

- void addOpaque (const char *name, mama_fid_t fid, const void *value, size_t size)

  Add an opaque field.

- void addOpaque (const MamaFieldDescriptor *fieldDesc, const void *value, size_t size)

  Add an opaque field.

- void addMsg (const char *name, mama_fid_t fid, MamaMsg *value)

  Add a MamaMsg object.

- void addMsg (const MamaFieldDescriptor *fieldDesc, MamaMsg *value)

  Add a MamaMsg object.

- void addVectorChar (const char *name, mama_fid_t fid, const char vectorValues[ ], size_t vectorLen)

  Add a vector of characters.

- void addVectorChar (const MamaFieldDescriptor *fieldDesc, const char vectorValues[ ], size_t vectorLen)

  Add a vector of characters.

- void addVectorI8 (const char *name, mama_fid_t fid, const mama_i8_t vectorValues[ ], size_t vectorLen)

  Add a vector of signed 8-bit integers.

- void addVectorI8 (const MamaFieldDescriptor *fieldDesc, const mama_i8_t vectorValues[ ], size_t vectorLen)

  Add a vector of signed 8-bit integers.

- void addVectorU8 (const char *name, mama_fid_t fid, const mama_u8_t vectorValues[ ], size_t vectorLen)

Add a vector of unsigned 8-bit integers.

- void addVectorU8 (const MamaFieldDescriptor *fieldDesc, const mama_u8_t vectorValues[ ], size_t vectorLen)

  Add a vector of unsigned 8-bit integers.

- void addVectorI16 (const char *name, mama_fid_t fid, const mama_i16_t vectorValues[ ], size_t vectorLen)

  Add a vector of signed 16-bit integers.

- void addVectorI16 (const MamaFieldDescriptor *fieldDesc, const mama_i16_t vectorValues[ ], size_t vectorLen)

  Add a vector of signed 16-bit integers.

- void addVectorU16 (const char *name, mama_fid_t fid, const mama_u16_t vectorValues[ ], size_t vectorLen)

  Add a vector of unsigned 16-bit integers.

- void addVectorU16 (const MamaFieldDescriptor *fieldDesc, const mama_u16_t vectorValues[ ], size_t vectorLen)

  Add a vector of unsigned 16-bit integers.

- void addVectorI32 (const char *name, mama_fid_t fid, const mama_i32_t vectorValues[ ], size_t vectorLen)

  Add a vector of signed 32-bit integers.

- void addVectorI32 (const MamaFieldDescriptor *fieldDesc, const mama_i32_t vectorValues[ ], size_t vectorLen)

  Add a vector of signed 32-bit integers.

- void addVectorU32 (const char *name, mama_fid_t fid, const mama_u32_t vectorValues[ ], size_t vectorLen)

  Add a vector of unsigned 32-bit integers.

- void addVectorU32 (const MamaFieldDescriptor *fieldDesc, const mama_u32_t vectorValues[ ], size_t vectorLen)

  Add a vector of unsigned 32-bit integers.

- void addVectorI64 (const char *name, mama_fid_t fid, const mama_i64_t vectorValues[ ], size_t vectorLen)

  Add a vector of signed 64-bit integers.

- void addVectorI64 (const MamaFieldDescriptor *fieldDesc, const mama_i64_t vectorValues[ ], size_t vectorLen)

Add a vector of signed 64-bit integers.

- void addVectorU64 (const char *name, mama_fid_t fid, const mama_u64_t vectorValues[ ], size_t vectorLen)

  Add a vector of unsigned 64-bit integers.

- void addVectorU64 (const MamaFieldDescriptor *fieldDesc, const mama_u64_t vectorValues[ ], size_t vectorLen)

  Add a vector of unsigned 64-bit integers.

- void addVectorF32 (const char *name, mama_fid_t fid, const mama_f32_t vectorValues[ ], size_t vectorLen)

  Add a vector of 32-bit floating point numbers.

- void addVectorF32 (const MamaFieldDescriptor *fieldDesc, const mama_f32_t vectorValues[ ], size_t vectorLen)

  Add a vector of unsigned 32-bit integers.

- void addVectorF64 (const char *name, mama_fid_t fid, const mama_f64_t vectorValues[ ], size_t vectorLen)

  Add a vector of 64-bit floating point numbers.

- void addVectorF64 (const MamaFieldDescriptor *fieldDesc, const mama_f64_t vectorValues[ ], size_t vectorLen)

  Add a vector of unsigned 64-bit integers.

- void addVectorString (const char *name, mama_fid_t fid, const char *vectorValues[ ], size_t vectorLen)

  Add a vector of strings.

- void addVectorString (const MamaFieldDescriptor *fieldDesc, const char *vectorValues[ ], size_t vectorLen)

  Add a vector of strings.

- void addVectorMsg (const char *name, mama_fid_t fid, MamaMsg *vectorValues[ ], size_t vectorLen)

  Add a vector of MamaMsg objects.

- void addVectorMsg (const MamaFieldDescriptor *fieldDesc, MamaMsg *vectorValues[ ], size_t vectorLen)

  Add a vector of MamaMsg objects.

- void updateBoolean (const char *name, mama_fid_t fid, bool value)

Update the value of an existing boolean field.

- void updateBoolean (const MamaFieldDescriptor *fieldDesc, bool value)

  Update the value of an existing boolean field.

- void updateChar (const char *name, mama_fid_t fid, const char value)

  Update the value of an existing char field.

- void updateChar (const MamaFieldDescriptor *fieldDesc, const char value)

  Update the value of an existing char field.

- void updateI8 (const char *name, mama_fid_t fid, const mama_i8_t value)

  Update the value of an existing byte field.

- void updateI8 (const MamaFieldDescriptor *fieldDesc, const mama_i8_t value)

  Update the value of an existing byte field.

- void updateU8 (const char *name, mama_fid_t fid, const mama_u8_t value)

  Update the value of an existing U8 field.

- void updateU8 (const MamaFieldDescriptor *fieldDesc, const mama_u8_t value)

  Update the value of an existing U8 field.

- void updateI16 (const char *name, mama_fid_t fid, const mama_i16_t value)

  Update the value of an existing short field.

- void updateI16 (const MamaFieldDescriptor *fieldDesc, const mama_i16_t value)

  Update the value of an existing short field.

- void updateU16 (const char *name, mama_fid_t fid, const mama_u16_t value)

  Update the value of an existing U16 field.

- void updateU16 (const MamaFieldDescriptor *fieldDesc, const mama_u16_t value)

  Update the value of an existing U16 field.

- void updateI32 (const char *name, mama_fid_t fid, const mama_i32_t value)

  Update the value of an existing integer field.

- void updateI32 (const MamaFieldDescriptor *fieldDesc, const mama_i32_t value)

  Update the value of an existing integer field.

- void updateU32 (const char *name, mama_fid_t fid, const mama_u32_t value)

  Update the value of an existing U32 field.

- void updateU32 (const MamaFieldDescriptor *fieldDesc, const mama_u32_t value)

  Update the value of an existing U32 field.

- void updateI64 (const char *name, mama_fid_t fid, const mama_i64_t value)

  Update the value of an existing long field.

- void updateI64 (const MamaFieldDescriptor *fieldDesc, const mama_i64_t value)

  Update the value of an existing long field.

- void updateU64 (const char *name, mama_fid_t fid, const mama_u64_t value)

  Update the value of an existing U64 field.

- void updateU64 (const MamaFieldDescriptor *fieldDesc, const mama_u64_t value)

  Update the value of an existing U64 field.

- void updateF32 (const char *name, mama_fid_t fid, const mama_f32_t value)

  Update the value of an existing F32 field.

- void updateF32 (const MamaFieldDescriptor *fieldDesc, const mama_f32_t value)

  Update the value of an existing F32 field.

- void updateF64 (const char *name, mama_fid_t fid, const mama_f64_t value)

  Update the value of an existing F64 field.

- void updateF64 (const MamaFieldDescriptor *fieldDesc, const mama_f64_t value)

  Update the value of an existing F64 field.

- void updateString (const char *name, mama_fid_t fid, const char *value)

  Update the value of an existing const char field.*

- void updateString (const MamaFieldDescriptor *fieldDesc, const char *value)

Update the value of an existing string field.

- void updateOpaque (const char *name, mama_fid_t fid, const void *value, size_t size)

  Update the value of an existing opaque field.

- void updateOpaque (const MamaFieldDescriptor *fieldDesc, const void *value, size_t size)

  Update the value of an existing opaque field.

- void updateDateTime (const char *name, mama_fid_t fid, const MamaDateTime &value)

  Update a date/time field.

- void updateDateTime (const MamaFieldDescriptor *fieldDesc, const MamaDateTime &value)

  Update the value of an existing date/time field.

- void updatePrice (const char *name, mama_fid_t fid, const MamaPrice &value)

  Update a price field.

- void updatePrice (const MamaFieldDescriptor *fieldDesc, const MamaPrice &value)

  Update the value of an existing price field.

- mamaMsgType getType (void) const

  Get the message type.

- const char * getMsgTypeName (void) const

  Get a human readable type name.

- mamaMsgStatus getStatus (void) const

  Get the msg status.

- const char * getMsgStatusString (void) const

  Get human readable message status.

- void iterateFields (MamaMsgFieldIterator &iterator, const MamaDictionary *dictionary, void *closure) const

  Iterate over all the fields.

- const char * toString () const

*Return a const char * representation the message.*

- void getFieldAsString (const char *name, mama_fid_t fid, char *result, size_t maxResultLen) const

  Obtain a string representation the field with the given fid.

- void getFieldAsString (const MamaFieldDescriptor *fieldDesc, char *result, size_t maxResultLen) const

  Obtain a string representation the field with the given fid.

- MamaMsgField * getField (const char *name, mama_fid_t fid) const

  Obtain a the mamaMsgField with the given fid.

- MamaMsgField * getField (const MamaFieldDescriptor *fieldDesc) const

  Obtain a the mamaMsgField with the given fid.

- bool tryField (const char *name, mama_fid_t fid) const

  Test for the presence of the MamaMsgField with the given fid.

- bool tryField (const MamaFieldDescriptor *fieldDesc) const

  Test for the presence of the MamaMsgField with the given field descriptor.

- bool tryField (const char *name, mama_fid_t fid, MamaMsgField *result) const

  Try to obtain the MamaMsgField with the given fid.

- bool tryField (const MamaFieldDescriptor *fieldDesc, MamaMsgField *result) const

  Try to obtain the MamaMsgField with the given field descriptor.

- bool tryFieldAsString (const char *name, mama_fid_t fid, char *result, size_t maxResultLen) const

  Try to obtain a string representation the field with the given fid.

- bool tryFieldAsString (const MamaFieldDescriptor *fieldDesc, char *result, size_t maxResultLen) const

  Try to obtain a string representation the field with the given fid.

- void getByteBuffer (const void *&buffer, size_t &bufferLength) const

  Get the underlying message as an array of bytes.

- void createForBridge (mamaBridge bridgeImpl)

  Create a MamaMsg.

- MamaMsg * detach (void)

  Normally the Mama API owns incoming mamaMsg objects and they go out of scope and are freed when the message callback returns.

- bool isFromInbox (void) const

  Whether this message is the result of a request needing a response.

- mama_seqnum_t getSeqNum (void) const
- bool getIsDefinitelyDuplicate (void) const

  Return true if this message is definitely a duplicate message.

- bool getIsPossiblyDuplicate (void) const

  Return true if this message is possibly a duplicate message.

- bool getIsPossiblyDelayed (void) const

  Return true if the message is possibly delayed.

- bool getIsDefinitelyDelayed (void) const

  Return true if the message is delayed.

- bool getIsOutOfSequence (void) const

  Return true when the FH sends the message out of sequence.

- bool setNewBuffer (void *buffer, mama_size_t size)

  Sets a new for an existing mamaMsg using the provided byte buffer.

- void * getNativeHandle (void) const

  Return the native middleware message handle.

- void createFromMsg (mamaMsg msg, bool destroyMsg=false) const

  Create the message from an existing mamaMsg.

- void setMsg (mamaMsg msg)

  Set the message to a different underlying C message.

- const mamaMsg & getUnderlyingMsg (void) const

  Return const reference to underlying mamaMsg.

- mamaMsg getUnderlyingMsg (void)

  Return the underlying mamaMsg (non const).

- mamaPayloadType getPayloadType (void) const

Return the type of the payload message (wombat message or, if using a non-wombat message payload, RV or FAST message).

- void * getNativeMsg (void)

  Get the native message structure for the underlying message.

- MamaMsgField & begin (MamaMsgIterator &theIterator) const

  Sets a iterator to be used with existing mamaMsg.

- mamaMsgReply getReplyHandle (void) const

  Get a copy of the reply Handle.

Static Public Member Functions

- static void destroyReplyHandle (mamaMsgReply replyHandle)

  Destroy a copied reply Handle.

7.24.1 Detailed Description

MAMA message representation.

Field identifiers must be greater than 0. A field identifier of 0 indicates that there is no unique FID and multiple fields with the same name may have FID == 0.

Field lookup proceeds in the following manner similar to TIBRV.

- If the fid supplied is non-zero, search for a field with the specified fid and return the field if it exists (the name is not validated). Otherwise throw a STATUS_-NOT_FOUND exception.
- If the fid supplied is 0, return the first field encountered with the supplied name or throw a STATUS_NOT_FOUND exception if no such field exists.

Get methods for numeric values may result in loss of information through either rounding or truncation when a larger data type is accessed as a smaller one. The result may be the same as the result of casting the larger value to the smaller. For example calling `getShort` for an integer field with a value greater than `Short.MAX_VALUE` might return `Short.MIN_VALUE`. It is also valid to throw a `ClassCastException` or other appropriate `RuntimeException`.

Since some message implementations may not natively support all data types, the behaviour may vary substantially. In creating and accessing messages the API's assume that the underlying values are stored in the smallest possible fields, and accesses them

accordingly. For this reason the minimal requirement is that the methods for accessing and creating fields support the full range of values appropriate for their type. How they deal with larger values should be irrelevant.

Author:

Michael Schonberg

7.24.2 Constructor & Destructor Documentation

7.24.2.1 Wombat::MamaMsg::~MamaMsg ()

7.24.2.2 Wombat::MamaMsg::MamaMsg (void)

7.24.2.3 Wombat::MamaMsg::MamaMsg (const MamaMsg & *mm*)

7.24.2.4 Wombat::MamaMsg::MamaMsg (WombatMsg & *wm*)

7.24.3 Member Function Documentation

7.24.3.1 void Wombat::MamaMsg::create (void)

Create the actual underlying wire format message.

This method will create a an underlying message native to the middleware being used. Tibrv: RV message LBM: Wombat Message TCP: Wombat Message

7.24.3.2 void Wombat::MamaMsg::createForPayload (const char *id*)

Create a mamaMsg.

Parameters:

id The identifier of the payload to be used.

7.24.3.3 void Wombat::MamaMsg::createForPayloadBridge (mamaPayloadBridge *payloadBridge*)

Create a mamaMsg.

Parameters:

id The payload bridge to be used.

7.24.3.4 void Wombat::MamaMsg::createForWombatMsg (void)

The underlying wire format for the message will be Wombat Message regardless of the middleware being used.

7.24.3.5 void Wombat::MamaMsg::createFromBuffer (const void * *buffer*, size_t *bufferLength*)

Create a MamaMsg from the provided byte buffer.

The application is responsible for destroying the message. In all cases the buffer is copied as the message is constructed.

Any transport differences are detailed below.

rv: The bufferLength parameter is not required.

Note: wombatmsg format is not currently supported on tibrv transports. At the moment a buffer containing the wire format for each of these transports is expected to be passed to the function.

Parameters:

buffer The byte array containing the wire format of the message

bufferLength The length, in bytes, of the supplied buffer

Returns:

mama_status The outcome of the operation

7.24.3.6 void Wombat::MamaMsg::createForBridgeFromBuffer (const void * *buffer*, size_t *bufferLength*, mamaBridge *bridge*)

Create a mamaMsg from the provided byte buffer using either the native format for the bridge (e.g.

TIB/RV messages for the TIB/RV bridge) or a wombatmsg. The function can determine from the buffer whether it is a wombatmsg or the native format for the transport being used. The application is responsible for destroying the message.

Parameters:

buffer The byte array containing the wire format of the message

bufferLength The length, in bytes, of the supplied buffer

bridge The bridge to use

7.24.3.7 void Wombat::MamaMsg::copy (const MamaMsg & *rhs*)

Copy the message from another, severing all links to the original message.

7.24.3.8 MamaMsg* Wombat::MamaMsg::getTempCopy ()

Get a temporary copy of the mamaMsg so to be able to modify the content.

If the message can be modified directly, the message itself is returned. If the message cannot be modified then only one copy is performed the first time this method is called and then the same copy is returned when this method is called again. The copy has the same life time of the original message.

Returns:

The message copy.

7.24.3.9 void Wombat::MamaMsg::applyMsg (const MamaMsg & *msg*)

7.24.3.10 void Wombat::MamaMsg::clear (void)

Clear the message.

All fields are removed.

7.24.3.11 size_t Wombat::MamaMsg::getNumFields (void) const

Returns the total number of fields in the message.

Sub-messages count as a single field.

Returns:

The number of fields in the message.

7.24.3.12 size_t Wombat::MamaMsg::getByteSize (void) const

Get the message size in bytes.

Supported with the following transports: tibrv LBT

Returns:

The number of bytes in the message

7.24.3.13 bool Wombat::MamaMsg::getBoolean (const char * *name*, mama_fid_t *fid*) const

Get a boolean field.

Parameters:

name The name

fid The field identifier

Returns:

The integer value.

7.24.3.14 bool Wombat::MamaMsg::getBoolean (const MamaFieldDescriptor * *fieldDesc*) const

Get a boolean field.

Parameters:

fieldDesc The field descriptor

Returns:

The integer value.

7.24.3.15 char Wombat::MamaMsg::getChar (const char * *name*, mama_fid_t *fid*) const

Get a char field.

Parameters:

name The name

fid The field identifier

Returns:

The integer value.

7.24.3.16 char Wombat::MamaMsg::getChar (const MamaFieldDescriptor * *fieldDesc*) const

Get a char field.

Parameters:

fieldDesc The field descriptor

Returns:

The integer value.

7.24.3.17 mama_i8_t Wombat::MamaMsg::getI8 (const char * *name*, mama_fid_t *fid*) const

Get an I8 field.

Parameters:

name The name

fid The field identifier

Returns:

The integer value.

7.24.3.18 mama_i8_t Wombat::MamaMsg::getI8 (const MamaFieldDescriptor * *fieldDesc*) const

Get a I8 field.

Parameters:

fieldDesc The field descriptor

Returns:

The integer value.

7.24.3.19 mama_u8_t Wombat::MamaMsg::getU8 (const char * *name*, mama_fid_t *fid*) const

Get a U8 field.

Parameters:

name The name

fid The field identifier

Returns:

The integer value.

7.24.3.20 mama_u8_t Wombat::MamaMsg::getU8 (const MamaFieldDescriptor * *fieldDesc*) const

Get a U8 field.

Parameters:

fieldDesc The field descriptor

Returns:

The integer value.

7.24.3.21 mama_i16_t Wombat::MamaMsg::getI16 (const char * *name*, mama_fid_t *fid*) const

Get an I16 field.

Parameters:

name The name

fid The field identifier

Returns:

The integer value.

7.24.3.22 mama_i16_t Wombat::MamaMsg::getI16 (const MamaFieldDescriptor * *fieldDesc*) const

Get an I16 field.

Parameters:

fieldDesc The field descriptor

Returns:

The integer value.

7.24.3.23 mama_u16_t Wombat::MamaMsg::getU16 (const char * *name*, mama_fid_t *fid*) const

Get a U16 field.

Parameters:

name The name

fid The field identifier

Returns:

The integer value.

7.24.3.24 mama_u16_t Wombat::MamaMsg::getU16 (const MamaFieldDescriptor * *fieldDesc*) const

Get an U16 field.

Parameters:

fieldDesc The field descriptor

Returns:

The integer value.

7.24.3.25 mama_i32_t Wombat::MamaMsg::getI32 (const char * *name*, mama_fid_t *fid*) const

Get an I32 field.

Parameters:

name The name

fid The field identifier

Returns:

The integer value.

7.24.3.26 mama_i32_t Wombat::MamaMsg::getI32 (const MamaFieldDescriptor * *fieldDesc*) const

Get a I32 field.

Parameters:

fieldDesc The field descriptor

Returns:

The integer value.

7.24.3.27 mama_u32_t Wombat::MamaMsg::getU32 (const char * *name*, mama_fid_t *fid*) const

Get a U32 field.

Parameters:

name The name

fid The field identifier

Returns:

The integer value.

7.24.3.28 mama_u32_t Wombat::MamaMsg::getU32 (const MamaFieldDescriptor * *fieldDesc*) const

Get a U32 field.

Parameters:

fieldDesc The field descriptor

Returns:

The integer value.

7.24.3.29 mama_i64_t Wombat::MamaMsg::getI64 (const char * *name*, mama_fid_t *fid*) const

Get a I64 field.

Parameters:

name The name

fid The field identifier.

Returns:

The field value as a long.

7.24.3.30 mama_i64_t Wombat::MamaMsg::getI64 (const MamaFieldDescriptor * *fieldDesc*) const

Get a I64 field.

Parameters:

fieldDesc The field descriptor

Returns:

The integer value.

7.24.3.31 mama_u64_t Wombat::MamaMsg::getU64 (const char * *name*, mama_fid_t *fid*) const

Get a U64 field.

Parameters:

name The name

fid The field identifier

Returns:

The integer value.

7.24.3.32 mama_u64_t Wombat::MamaMsg::getU64 (const MamaFieldDescriptor * *fieldDesc*) const

Get a U64 field.

Parameters:

fieldDesc The field descriptor

Returns:

The integer value.

7.24.3.33 mama_f32_t Wombat::MamaMsg::getF32 (const char * *name*, mama_fid_t *fid*) const

Get a f32 field.

Parameters:

name The name.

fid The field identifier.

Returns:

The double value.

7.24.3.34 mama_f32_t Wombat::MamaMsg::getF32 (const MamaFieldDescriptor * *fieldDesc*) const

Get a f32 field.

Parameters:

fieldDesc The field descriptor

Returns:

The double value.

7.24.3.35 mama_f64_t Wombat::MamaMsg::getF64 (const char * *name*, mama_fid_t *fid*) const

Get a f64 field.

Parameters:

name The name.

fid The field identifier.

Returns:

The double value.

7.24.3.36 mama_f64_t Wombat::MamaMsg::getF64 (const MamaFieldDescriptor * *fieldDesc*) const

Get a f64 field.

Parameters:

fieldDesc The field descriptor

Returns:

The double value.

7.24.3.37 const char* Wombat::MamaMsg::getString (const char * *name*, mama_fid_t *fid*) const

Get a const char* field.

Parameters:

name The field name.

fid The field identifier.

Returns:

The string value. The return value points to the string inside the mamaMsg object. This piece of memory is owned by the object and does not need to be explicitly freed.

7.24.3.38 const char* Wombat::MamaMsg::getString (const MamaFieldDescriptor * *fieldDesc*) const

Get a const char* field.

Parameters:

fieldDesc Pointer to the field descriptor

Returns:

The string value. The return value points to the string inside the mamaMsg object. This piece of memory is owned by the object and does not need to be explicitly freed.

7.24.3.39 const void* Wombat::MamaMsg::getOpaque (const char * *name*, mama_fid_t *fid*, size_t & *size*) const

Get an opaque field.

Parameters:

name The field name.

fid The field identifier.

→ ***size*** The opaque size in bytes.

Returns:

the string value.

7.24.3.40 const void* Wombat::MamaMsg::getOpaque (const MamaFieldDescriptor * *fieldDesc*, size_t & *size*) const

Get a const char* field.

Parameters:

fieldDesc The field descriptor

→ ***size*** The opaque size in bytes.

Returns:

The double value.

7.24.3.41 void Wombat::MamaMsg::getDateTime (const char * *name*, mama_fid_t *fid*, MamaDateTime & *result*) const

Get a MamaDateTime field.

Parameters:

name The field name.

fid The field identifier.

result (out) the date/time.

7.24.3.42 void Wombat::MamaMsg::getDateTime (const MamaFieldDescriptor * *fieldDesc*, MamaDateTime & *result*) const

Get a MamaDateTime field.

Parameters:

fieldDesc The field descriptor

result (out) the date/time.

7.24.3.43 void Wombat::MamaMsg::getPrice (const char * *name*, mama_fid_t *fid*, MamaPrice & *result*) const

Get a MamaPrice field.

Parameters:

name The field name.

fid The field identifier.

result (out) the date/time.

7.24.3.44 void Wombat::MamaMsg::getPrice (const MamaFieldDescriptor * *fieldDesc*, MamaPrice & *result*) const

Get a MamaPrice field.

Parameters:

fieldDesc The field descriptor

result (out) the date/time.

7.24.3.45 const MamaMsg* Wombat::MamaMsg::getMsg (const char * *name*, mama_fid_t *fid*) const

Get a submessage field.

Parameters:

name The field name.

fid The field identifier.

Returns:

The submessage.

7.24.3.46 const MamaMsg* Wombat::MamaMsg::getMsg (const MamaFieldDescriptor * *fieldDesc*) const

Get a submessage field.

Parameters:

fieldDesc The field descriptor

Returns:

The vector.

7.24.3.47 const char* Wombat::MamaMsg::getVectorChar (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of characters.

(Note: prefer using string fields or opaque fields over a vector of characters.)

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.48 const char* Wombat::MamaMsg::getVectorChar (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of characters.

(Note: prefer using string fields or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.49 const mama_i8_t* Wombat::MamaMsg::getVectorI8 (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of signed 8-bit integers.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.50 const mama_i8_t* Wombat::MamaMsg::getVectorI8 (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of signed 8-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.51 const mama_u8_t* Wombat::MamaMsg::getVectorU8 (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of unsigned 8-bit integers.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.52 const mama_u8_t* Wombat::MamaMsg::getVectorU8 (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of unsigned 8-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.53 const mama_i16_t* Wombat::MamaMsg::getVectorI16 (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of signed 16-bit integers.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.54 const mama_i16_t* Wombat::MamaMsg::getVectorI16 (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of signed 16-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.55 const mama_u16_t* Wombat::MamaMsg::getVectorU16 (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of unsigned 16-bit integers.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.56 const mama_u16_t* Wombat::MamaMsg::getVectorU16 (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of unsigned 16-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.57 const mama_i32_t* Wombat::MamaMsg::getVectorI32 (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of signed 32-bit integers.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.58 const mama_i32_t* Wombat::MamaMsg::getVectorI32 (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of signed 32-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.59 const mama_u32_t* Wombat::MamaMsg::getVectorU32 (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of unsigned 32-bit integers.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.60 const mama_u32_t* Wombat::MamaMsg::getVectorU32 (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of unsigned 32-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.61 const mama_i64_t* Wombat::MamaMsg::getVectorI64 (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of signed 64-bit integers.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.62 const mama_i64_t* Wombat::MamaMsg::getVectorI64 (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of signed 64-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.63 const mama_u64_t* Wombat::MamaMsg::getVectorU64 (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of unsigned 64-bit integers.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.64 const mama_u64_t* Wombat::MamaMsg::getVectorU64 (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of unsigned 64-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.65 const mama_f32_t* Wombat::MamaMsg::getVectorF32 (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of 32-bit floating point numbers.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.66 const mama_f32_t* Wombat::MamaMsg::getVectorF32 (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of 32-bit floating point numbers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.67 const mama_f64_t* Wombat::MamaMsg::getVectorF64 (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of 64-bit floating point numbers.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.68 const mama_f64_t* Wombat::MamaMsg::getVectorF64 (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of 64-bit floating point numbers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.69 const MamaMsg** Wombat::MamaMsg::getVectorMsg (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of submessages field.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.70 const MamaMsg** Wombat::MamaMsg::getVectorMsg (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of submessages.

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.71 const char** Wombat::MamaMsg::getVectorString (const char * *name*, mama_fid_t *fid*, size_t & *resultLen*) const

Get a vector of strings field.

The vector is deleted with the message and overwritten by subsequent calls to getVectorString.

Parameters:

name The field name.

fid The field identifier.

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.72 const char** Wombat::MamaMsg::getVectorString (const MamaFieldDescriptor * *fieldDesc*, size_t & *resultLen*) const

Get a vector of submessages.

The vector is deleted with the message and overwritten by subsequent calls to getVectorString.

Parameters:

fieldDesc The field descriptor

resultLen (out) the size of the vector.

Returns:

The vector.

7.24.3.73 bool Wombat::MamaMsg::tryBoolean (const char * *name*, mama_fid_t *fid*, bool & *result*) const

Try to get a boolean field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.74 bool Wombat::MamaMsg::tryBoolean (const MamaFieldDescriptor * *field*, bool & *result*) const

Try to get a boolean field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.75 bool Wombat::MamaMsg::tryChar (const char * *name*, mama_fid_t *fid*, char & *result*) const

Try to get a char field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.76 bool Wombat::MamaMsg::tryChar (const MamaFieldDescriptor * *field*, char & *result*) const

Try to get a char field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.77 bool Wombat::MamaMsg::tryI8 (const char * *name*, mama_fid_t *fid*, mama_i8_t & *result*) const

Try to get an unsigned 8 bit integer field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.78 bool Wombat::MamaMsg::tryI8 (const MamaFieldDescriptor * *field*, mama_i8_t & *result*) const

Try to get an unsigned 8 bit field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.79 bool Wombat::MamaMsg::tryU8 (const char * *name*, mama_fid_t *fid*, mama_u8_t & *result*) const

Try to get an unsigned 8 bit integer field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.80 bool Wombat::MamaMsg::tryU8 (const MamaFieldDescriptor * *field*, mama_u8_t & *result*) const

Try to get an unsigned 8 bit field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.81 bool Wombat::MamaMsg::tryI16 (const char * *name*, mama_fid_t *fid*, mama_i16_t & *result*) const

Try to get a signed 16 bit integer field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.82 bool Wombat::MamaMsg::tryI16 (const MamaFieldDescriptor * *field*, mama_i16_t & *result*) const

Try to get a signed 16 bit field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.83 bool Wombat::MamaMsg::tryU16 (const char * *name*, mama_fid_t *fid*, mama_u16_t & *result*) const

Try to get an unsigned 16 bit integer field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.84 bool Wombat::MamaMsg::tryU16 (const MamaFieldDescriptor * *field*, mama_u16_t & *result*) const

Try to get an unsigned 16 bit field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.85 bool Wombat::MamaMsg::tryI32 (const char * *name*, mama_fid_t *fid*, mama_i32_t & *result*) const

Try to get a signed 32 bit integer field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.86 bool Wombat::MamaMsg::tryI32 (const MamaFieldDescriptor * *field*, mama_i32_t & *result*) const

Try to get a signed 32 bit integer field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.87 bool Wombat::MamaMsg::tryU32 (const char * *name*, mama_fid_t *fid*, mama_u32_t & *result*) const

Try to get an unsigned 32 bit integer field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.88 bool Wombat::MamaMsg::tryU32 (const MamaFieldDescriptor * *field*, mama_u32_t & *result*) const

Try to get an unsigned 32 bit integer field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.89 bool Wombat::MamaMsg::tryI64 (const char * *name*, mama_fid_t *fid*, mama_i64_t & *result*) const

Try to get a signed 64 bit field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.90 bool Wombat::MamaMsg::tryI64 (const MamaFieldDescriptor * *field*, mama_i64_t & *result*) const

Try to get a signed 64 bit field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.91 bool Wombat::MamaMsg::tryU64 (const char * *name*, mama_fid_t *fid*, mama_u64_t & *result*) const

Try to get an unsigned 64 bit field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.92 bool Wombat::MamaMsg::tryU64 (const MamaFieldDescriptor * *field*, mama_u64_t & *result*) const

Try to get an unsigned 64 bit field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.93 bool Wombat::MamaMsg::tryF32 (const char * *name*, mama_fid_t *fid*, mama_f32_t & *result*) const

Try to get a f32 field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.94 bool Wombat::MamaMsg::tryF32 (const MamaFieldDescriptor * *field*, mama_f32_t & *result*) const

Try to get a f32 field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.95 bool Wombat::MamaMsg::tryF64 (const char * *name*, mama_fid_t *fid*, mama_f64_t & *result*) const

Try to get a f64 field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.96 bool Wombat::MamaMsg::tryF64 (const MamaFieldDescriptor * *field*, mama_f64_t & *result*) const

Try to get a f64 field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.97 bool Wombat::MamaMsg::tryString (const char * *name*, mama_fid_t *fid*, const char *&* *result*) const

Try to get a string field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

Returns:

Whether the field was present.

7.24.3.98 bool Wombat::MamaMsg::tryString (const MamaFieldDescriptor * *field*, const char *&* *result*) const

Try to get a string field.

Parameters:

result The result (not modified if field not present) const

field The field to try

Returns:

Whether the field was present.

7.24.3.99 bool Wombat::MamaMsg::tryDateTime (const char * *name*, mama_fid_t *fid*, MamaDateTime & *result*) const

Try to get a date/time field.

Parameters:

name The field name

fid The field identifier

result The result (not modified if field not present) const

Returns:

Whether the field was present.

7.24.3.100 bool Wombat::MamaMsg::tryDateTime (const MamaFieldDescriptor * *field*, MamaDateTime & *result*) const

Try to get a date/time field.

Parameters:

field The field to try

result The result (not modified if field not present) const

Returns:

Whether the field was present.

7.24.3.101 bool Wombat::MamaMsg::tryPrice (const char * *name*, mama_fid_t *fid*, MamaPrice & *result*) const

Try to get a price field.

Parameters:

name The field name

fid The field identifier

result The result (not modified if field not present) const

Returns:

Whether the field was present.

7.24.3.102 bool Wombat::MamaMsg::tryPrice (const MamaFieldDescriptor * *field*, MamaPrice & *result*) const

Try to get a price field.

Parameters:

field The field to try

result The result (not modified if field not present) const

Returns:

Whether the field was present.

7.24.3.103 bool Wombat::MamaMsg::tryMsg (const char * *name*, mama_fid_t *fid*, const MamaMsg *& *result*) const

Try to get a submessage field.

Parameters:

name The field name

fid The field identifier

result The result

Returns:

Whether the field was present.

7.24.3.104 bool Wombat::MamaMsg::tryMsg (const MamaFieldDescriptor * *field*, const MamaMsg *& *result*) const

Try to get a submessage field.

Parameters:

field The field to try

result The result

Returns:

Whether the field was present.

7.24.3.105 bool Wombat::MamaMsg::tryOpaque (const char * *name*, mama_fid_t *fid*, const void *& *result*, size_t & *size*) const

Try to get a string field.

Parameters:

result The result (not modified if field not present) const

name The field name

fid The field identifier

size (out) The size of the opaque in bytes.

Returns:

Whether the field was present.

7.24.3.106 bool Wombat::MamaMsg::tryOpaque (const MamaFieldDescriptor * *field*, const void *& *result*, size_t & *size*) const

Try to get a string field.

Parameters:

result The result (not modified if field not present) const

field The field to try

size (out) The size of the opaque in bytes.

Returns:

Whether the field was present.

7.24.3.107 bool Wombat::MamaMsg::tryVectorChar (const char * *name*, mama_fid_t *fid*, const char *& *result*, size_t & *resultLen*) const

Try to get a vector of characters.

(Note: prefer using string fields or opaque fields over a vector of characters.)

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.108 bool Wombat::MamaMsg::tryVectorChar (const MamaFieldDescriptor * *fieldDesc*, const char *& *result*, size_t & *resultLen*) const

Try to get a vector of characters.

(Note: prefer using string fields or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.109 bool Wombat::MamaMsg::tryVectorI8 (const char * *name*, mama_fid_t *fid*, const mama_i8_t *& *result*, size_t & *resultLen*) const

Try to get a vector of signed 8-bit integers.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.110 bool Wombat::MamaMsg::tryVectorI8 (const MamaFieldDescriptor * *fieldDesc*, const mama_i8_t *& *result*, size_t & *resultLen*) const

Try to get a vector of signed 8-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.111 bool Wombat::MamaMsg::tryVectorU8 (const char * *name*, mama_fid_t *fid*, const mama_u8_t *& *result*, size_t & *resultLen*) const

Try to get a vector of unsigned 8-bit integers.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.112 bool Wombat::MamaMsg::tryVectorU8 (const MamaFieldDescriptor * *fieldDesc*, const mama_u8_t *& *result*, size_t & *resultLen*) const

Try to get a vector of unsigned 8-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.113 bool Wombat::MamaMsg::tryVectorI16 (const char * *name*, mama_fid_t *fid*, const mama_i16_t *& *result*, size_t & *resultLen*) const

Try to get a vector of signed 16-bit integers.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.114 bool Wombat::MamaMsg::tryVectorI16 (const MamaFieldDescriptor * *fieldDesc*, const mama_i16_t *& *result*, size_t & *resultLen*) const

Try to get a vector of signed 16-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.115 bool Wombat::MamaMsg::tryVectorU16 (const char * *name*, mama_fid_t *fid*, const mama_u16_t *& *result*, size_t & *resultLen*) const

Try to get a vector of unsigned 16-bit integers.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.116 bool Wombat::MamaMsg::tryVectorU16 (const MamaFieldDescriptor * *fieldDesc*, const mama_u16_t *& *result*, size_t & *resultLen*) const

Try to get a vector of unsigned 16-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.117 bool Wombat::MamaMsg::tryVectorI32 (const char * *name*, mama_fid_t *fid*, const mama_i32_t *& *result*, size_t & *resultLen*) const

Try to get a vector of signed 32-bit integers.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.118 bool Wombat::MamaMsg::tryVectorI32 (const MamaFieldDescriptor * *fieldDesc*, const mama_i32_t *& *result*, size_t & *resultLen*) const

Try to get a vector of signed 32-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.119 bool Wombat::MamaMsg::tryVectorU32 (const char * *name*, mama_fid_t *fid*, const mama_u32_t *& *result*, size_t & *resultLen*) const

Try to get a vector of unsigned 32-bit integers.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.120 bool Wombat::MamaMsg::tryVectorU32 (const MamaFieldDescriptor * *fieldDesc*, const mama_u32_t *& *result*, size_t & *resultLen*) const

Try to get a vector of unsigned 32-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.121 bool Wombat::MamaMsg::tryVectorI64 (const char * *name*, mama_fid_t *fid*, const mama_i64_t *& *result*, size_t & *resultLen*) const

Try to get a vector of signed 64-bit integers.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.122 bool Wombat::MamaMsg::tryVectorI64 (const MamaFieldDescriptor * *fieldDesc*, const mama_i64_t *& *result*, size_t & *resultLen*) const

Try to get a vector of signed 64-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.123 bool Wombat::MamaMsg::tryVectorU64 (const char * *name*, mama_fid_t *fid*, const mama_u64_t *& *result*, size_t & *resultLen*) const

Try to get a vector of unsigned 64-bit integers.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.124 bool Wombat::MamaMsg::tryVectorU64 (const MamaFieldDescriptor * *fieldDesc*, const mama_u64_t *& *result*, size_t & *resultLen*) const

Try to get a vector of unsigned 64-bit integers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.125 bool Wombat::MamaMsg::tryVectorF32 (const char * *name*, mama_fid_t *fid*, const mama_f32_t *& *result*, size_t & *resultLen*) const

Try to get a vector of 32-bit floating point numbers.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.126 bool Wombat::MamaMsg::tryVectorF32 (const MamaFieldDescriptor * *fieldDesc*, const mama_f32_t *& *result*, size_t & *resultLen*) const

Try to get a vector of 32-bit floating point numbers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.127 bool Wombat::MamaMsg::tryVectorF64 (const char * *name*, mama_fid_t *fid*, const mama_f64_t *& *result*, size_t & *resultLen*) const

Try to get a vector of 64-bit floating point numbers.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.128 bool Wombat::MamaMsg::tryVectorF64 (const MamaFieldDescriptor * *fieldDesc*, const mama_f64_t *& *result*, size_t & *resultLen*) const

Try to get a vector of 64-bit floating point numbers.

or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.129 bool Wombat::MamaMsg::tryVectorString (const char * *name*, mama_fid_t *fid*, const char **& *result*, size_t & *resultLen*) const

Try to get a vector of strings.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.130 bool Wombat::MamaMsg::tryVectorString (const MamaFieldDescriptor * *fieldDesc*, const char **& *result*, size_t & *resultLen*) const

Try to get a vector of strings.

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.131 bool Wombat::MamaMsg::tryVectorMsg (const char * *name*, mama_fid_t *fid*, const MamaMsg **& *result*, size_t & *resultLen*) const

Try to get a vector of submessages field.

Parameters:

name The field name.

fid The field identifier.

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.132 bool Wombat::MamaMsg::tryVectorMsg (const MamaFieldDescriptor * *fieldDesc*, const MamaMsg **& *result*, size_t & *resultLen*) const

Try to get a vector of submessages.

Parameters:

fieldDesc The field descriptor

result (out) The vector.

resultLen (out) The size of the vector.

Returns:

Whether the field was present.

7.24.3.133 void Wombat::MamaMsg::addBoolean (const char * *name*, mama_fid_t *fid*, bool *value*)

Add a new boolean field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.134 void Wombat::MamaMsg::addBoolean (const MamaFieldDescriptor * *fieldDesc*, bool *value*)

Add a new boolean field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.135 void Wombat::MamaMsg::addChar (const char * *name*, mama_fid_t *fid*, char *value*)

Add a new char field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.136 void Wombat::MamaMsg::addChar (const MamaFieldDescriptor * *fieldDesc*, char *value*)

Add a new char field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.137 void Wombat::MamaMsg::addI8 (const char * *name*, mama_fid_t *fid*, mama_i8_t *value*)

Add a new I8 field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.138 void Wombat::MamaMsg::addI8 (const MamaFieldDescriptor * *fieldDesc*, mama_i8_t *value*)

Add a new I8 field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.139 void Wombat::MamaMsg::addI16 (const char * *name*, mama_fid_t *fid*, mama_i16_t *value*)

Add a new I16 field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.140 void Wombat::MamaMsg::addI16 (const MamaFieldDescriptor * *fieldDesc*, mama_i16_t *value*)

Add a new I16 field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.141 void Wombat::MamaMsg::addI32 (const char * *name*, mama_fid_t *fid*, mama_i32_t *value*)

Add a new I32 field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.142 void Wombat::MamaMsg::addI32 (const MamaFieldDescriptor * *fieldDesc*, mama_i32_t *value*)

Add a new I32 field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.143 void Wombat::MamaMsg::addI64 (const char * *name*, mama_fid_t *fid*, mama_i64_t *value*)

Add a new I64 field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.144 void Wombat::MamaMsg::addI64 (const MamaFieldDescriptor * *fieldDesc*, mama_i64_t *value*)

Add a new I64 field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.145 void Wombat::MamaMsg::addU8 (const char * *name*, mama_fid_t *fid*, mama_u8_t *value*)

Add a new byte (U8) const field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.146 void Wombat::MamaMsg::addU8 (const MamaFieldDescriptor * *fieldDesc*, mama_u8_t *value*)

Add a new U8 field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.147 void Wombat::MamaMsg::addU16 (const char * *name*, mama_fid_t *fid*, mama_u16_t *value*)

Add a new short (U16) const field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.148 void Wombat::MamaMsg::addU16 (const MamaFieldDescriptor * *fieldDesc*, mama_u16_t *value*)

Add a new U16 field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.149 void Wombat::MamaMsg::addU32 (const char * *name*, mama_fid_t *fid*, mama_u32_t *value*)

Add a new int (U32) const field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.150 void Wombat::MamaMsg::addU32 (const MamaFieldDescriptor * *fieldDesc*, mama_u32_t *value*)

Add a new U32 field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.151 void Wombat::MamaMsg::addU64 (const char * *name*, mama_fid_t *fid*, mama_u64_t *value*)

Add a new int (U64) const field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.152 void Wombat::MamaMsg::addU64 (const MamaFieldDescriptor * *fieldDesc*, mama_u64_t *value*)

Add a new U64 field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.153 void Wombat::MamaMsg::addF32 (const char * *name*, mama_fid_t *fid*, mama_f32_t *value*)

Add a new F32 field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.154 void Wombat::MamaMsg::addF32 (const MamaFieldDescriptor * *fieldDesc*, mama_f32_t *value*)

Add a new F32 field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.155 void Wombat::MamaMsg::addF64 (const char * *name*, mama_fid_t *fid*, mama_f64_t *value*)

Add a new F64 field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.156 void Wombat::MamaMsg::addF64 (const MamaFieldDescriptor * *fieldDesc*, mama_f64_t *value*)

Add a new F64 field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.157 void Wombat::MamaMsg::addString (const char * *name*, mama_fid_t *fid*, const char * *value*)

Add a const char* field.

Parameters:

name The name.

value The value.

fid The field identifier.

7.24.3.158 void Wombat::MamaMsg::addString (const MamaFieldDescriptor * *fieldDesc*, const char * *value*)

Add a new const char* field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.159 void Wombat::MamaMsg::addDateTime (const char * *name*, mama_fid_t *fid*, const MamaDateTime & *value*)

Add a date/time field.

Parameters:

name The field name.

fid The field identifier.

value The value.

7.24.3.160 void Wombat::MamaMsg::addDateTime (const MamaFieldDescriptor * *fieldDesc*, const MamaDateTime & *value*)

Add a new date/time field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.161 void Wombat::MamaMsg::addPrice (const char * *name*, mama_fid_t *fid*, const MamaPrice & *value*)

Add a price field.

Parameters:

name The field name.

fid The field identifier.

value The value.

7.24.3.162 void Wombat::MamaMsg::addPrice (const MamaFieldDescriptor * *fieldDesc*, const MamaPrice & *value*)

Add a new price field.

Parameters:

fieldDesc The field descriptor

value The value.

7.24.3.163 void Wombat::MamaMsg::addOpaque (const char * *name*, mama_fid_t *fid*, const void * *value*, size_t *size*)

Add an opaque field.

Parameters:

name The name.

value The value.

fid The field identifier.

size the size of the opaque field, in bytes

7.24.3.164 void Wombat::MamaMsg::addOpaque (const MamaFieldDescriptor * *fieldDesc*, const void * *value*, size_t *size*)

Add an opaque field.

Parameters:

fieldDesc The field descriptor

value The value.

size the size of the opaque field, in bytes

7.24.3.165 void Wombat::MamaMsg::addMsg (const char * *name*, mama_fid_t *fid*, MamaMsg * *value*)

Add a MamaMsg object.

Parameters:

name The field name.

fid The field identifier.

value the value to add

7.24.3.166 void Wombat::MamaMsg::addMsg (const MamaFieldDescriptor * *fieldDesc*, MamaMsg * *value*)

Add a MamaMsg object.

Parameters:

fieldDesc The field descriptor

value the value to add

7.24.3.167 void Wombat::MamaMsg::addVectorChar (const char * *name*, mama_fid_t *fid*, const char *vectorValues*[ ], size_t *vectorLen*)

Add a vector of characters.

(Note: prefer using string fields or opaque fields over a vector of characters.)

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.168 void Wombat::MamaMsg::addVectorChar (const MamaFieldDescriptor * *fieldDesc*, const char *vectorValues*[ ], size_t *vectorLen*)

Add a vector of characters.

(Note: prefer using string fields or opaque fields over a vector of characters.)

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.169 void Wombat::MamaMsg::addVectorI8 (const char * *name*, mama_fid_t *fid*, const mama_i8_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of signed 8-bit integers.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.170 void Wombat::MamaMsg::addVectorI8 (const MamaFieldDescriptor * *fieldDesc*, const mama_i8_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of signed 8-bit integers.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.171 void Wombat::MamaMsg::addVectorU8 (const char * *name*, mama_fid_t *fid*, const mama_u8_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of unsigned 8-bit integers.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.172 void Wombat::MamaMsg::addVectorU8 (const MamaFieldDescriptor * *fieldDesc*, const mama_u8_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of unsigned 8-bit integers.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.173 void Wombat::MamaMsg::addVectorI16 (const char * *name*, mama_fid_t *fid*, const mama_i16_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of signed 16-bit integers.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.174 void Wombat::MamaMsg::addVectorI16 (const MamaFieldDescriptor * *fieldDesc*, const mama_i16_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of signed 16-bit integers.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.175 void Wombat::MamaMsg::addVectorU16 (const char * *name*, mama_fid_t *fid*, const mama_u16_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of unsigned 16-bit integers.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.176 void Wombat::MamaMsg::addVectorU16 (const MamaFieldDescriptor * *fieldDesc*, const mama_u16_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of unsigned 16-bit integers.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.177 void Wombat::MamaMsg::addVectorI32 (const char * *name*, mama_fid_t *fid*, const mama_i32_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of signed 32-bit integers.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.178 void Wombat::MamaMsg::addVectorI32 (const MamaFieldDescriptor * *fieldDesc*, const mama_i32_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of signed 32-bit integers.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.179 void Wombat::MamaMsg::addVectorU32 (const char * *name*, mama_fid_t *fid*, const mama_u32_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of unsigned 32-bit integers.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.180 void Wombat::MamaMsg::addVectorU32 (const MamaFieldDescriptor * *fieldDesc*, const mama_u32_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of unsigned 32-bit integers.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.181 void Wombat::MamaMsg::addVectorI64 (const char * *name*, mama_fid_t *fid*, const mama_i64_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of signed 64-bit integers.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.182 void Wombat::MamaMsg::addVectorI64 (const MamaFieldDescriptor * *fieldDesc*, const mama_i64_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of signed 64-bit integers.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.183 void Wombat::MamaMsg::addVectorU64 (const char * *name*, mama_fid_t *fid*, const mama_u64_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of unsigned 64-bit integers.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.184 void Wombat::MamaMsg::addVectorU64 (const MamaFieldDescriptor * *fieldDesc*, const mama_u64_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of unsigned 64-bit integers.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.185 void Wombat::MamaMsg::addVectorF32 (const char * *name*, mama_fid_t *fid*, const mama_f32_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of 32-bit floating point numbers.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.186 void Wombat::MamaMsg::addVectorF32 (const MamaFieldDescriptor * *fieldDesc*, const mama_f32_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of unsigned 32-bit integers.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.187 void Wombat::MamaMsg::addVectorF64 (const char * *name*, mama_fid_t *fid*, const mama_f64_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of 64-bit floating point numbers.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.188 void Wombat::MamaMsg::addVectorF64 (const MamaFieldDescriptor * *fieldDesc*, const mama_f64_t *vectorValues*[ ], size_t *vectorLen*)

Add a vector of unsigned 64-bit integers.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.189 void Wombat::MamaMsg::addVectorString (const char * *name*, mama_fid_t *fid*, const char * *vectorValues*[ ], size_t *vectorLen*)

Add a vector of strings.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.190 void Wombat::MamaMsg::addVectorString (const MamaFieldDescriptor * *fieldDesc*, const char * *vectorValues*[ ], size_t *vectorLen*)

Add a vector of strings.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.191 void Wombat::MamaMsg::addVectorMsg (const char * *name*, mama_fid_t *fid*, MamaMsg * *vectorValues*[ ], size_t *vectorLen*)

Add a vector of MamaMsg objects.

Parameters:

name The field name.

fid The field identifier.

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.192 void Wombat::MamaMsg::addVectorMsg (const MamaFieldDescriptor * *fieldDesc*, MamaMsg * *vectorValues*[ ], size_t *vectorLen*)

Add a vector of MamaMsg objects.

Parameters:

fieldDesc The field descriptor

vectorValues The vector values.

vectorLen The size of the vector.

7.24.3.193 void Wombat::MamaMsg::updateBoolean (const char * *name*, mama_fid_t *fid*, bool *value*)

Update the value of an existing boolean field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.194 void Wombat::MamaMsg::updateBoolean (const MamaFieldDescriptor * *fieldDesc*, bool *value*)

Update the value of an existing boolean field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.195 void Wombat::MamaMsg::updateChar (const char * *name*, mama_fid_t *fid*, const char *value*)

Update the value of an existing char field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.196 void Wombat::MamaMsg::updateChar (const MamaFieldDescriptor * *fieldDesc*, const char *value*)

Update the value of an existing char field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.197 void Wombat::MamaMsg::updateI8 (const char * *name*, mama_fid_t *fid*, const mama_i8_t *value*)

Update the value of an existing byte field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.198 void Wombat::MamaMsg::updateI8 (const MamaFieldDescriptor * *fieldDesc*, const mama_i8_t *value*)

Update the value of an existing byte field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.199 void Wombat::MamaMsg::updateU8 (const char * *name*, mama_fid_t *fid*, const mama_u8_t *value*)

Update the value of an existing U8 field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.200 void Wombat::MamaMsg::updateU8 (const MamaFieldDescriptor * *fieldDesc*, const mama_u8_t *value*)

Update the value of an existing U8 field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.201 void Wombat::MamaMsg::updateI16 (const char * *name*, mama_fid_t *fid*, const mama_i16_t *value*)

Update the value of an existing short field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.202 void Wombat::MamaMsg::updateI16 (const MamaFieldDescriptor * *fieldDesc*, const mama_i16_t *value*)

Update the value of an existing short field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.203 void Wombat::MamaMsg::updateU16 (const char * *name*, mama_fid_t *fid*, const mama_u16_t *value*)

Update the value of an existing U16 field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.204 void Wombat::MamaMsg::updateU16 (const MamaFieldDescriptor * *fieldDesc*, const mama_u16_t *value*)

Update the value of an existing U16 field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.205 void Wombat::MamaMsg::updateI32 (const char * *name*, mama_fid_t *fid*, const mama_i32_t *value*)

Update the value of an existing integer field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.206 void Wombat::MamaMsg::updateI32 (const MamaFieldDescriptor * *fieldDesc*, const mama_i32_t *value*)

Update the value of an existing integer field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.207 void Wombat::MamaMsg::updateU32 (const char * *name*, mama_fid_t *fid*, const mama_u32_t *value*)

Update the value of an existing U32 field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.208 void Wombat::MamaMsg::updateU32 (const MamaFieldDescriptor * *fieldDesc*, const mama_u32_t *value*)

Update the value of an existing U32 field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.209 void Wombat::MamaMsg::updateI64 (const char * *name*, mama_fid_t *fid*, const mama_i64_t *value*)

Update the value of an existing long field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.210 void Wombat::MamaMsg::updateI64 (const MamaFieldDescriptor * *fieldDesc*, const mama_i64_t *value*)

Update the value of an existing long field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.211 void Wombat::MamaMsg::updateU64 (const char * *name*, mama_fid_t *fid*, const mama_u64_t *value*)

Update the value of an existing U64 field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.212 void Wombat::MamaMsg::updateU64 (const MamaFieldDescriptor * *fieldDesc*, const mama_u64_t *value*)

Update the value of an existing U64 field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.213 void Wombat::MamaMsg::updateF32 (const char * *name*, mama_fid_t *fid*, const mama_f32_t *value*)

Update the value of an existing F32 field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.214 void Wombat::MamaMsg::updateF32 (const MamaFieldDescriptor * *fieldDesc*, const mama_f32_t *value*)

Update the value of an existing F32 field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.215 void Wombat::MamaMsg::updateF64 (const char * *name*, mama_fid_t *fid*, const mama_f64_t *value*)

Update the value of an existing F64 field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.216 void Wombat::MamaMsg::updateF64 (const MamaFieldDescriptor * *fieldDesc*, const mama_f64_t *value*)

Update the value of an existing F64 field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.217 void Wombat::MamaMsg::updateString (const char * *name*, mama_fid_t *fid*, const char * *value*)

Update the value of an existing const char* field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

7.24.3.218 void Wombat::MamaMsg::updateString (const MamaFieldDescriptor * *fieldDesc*, const char * *value*)

Update the value of an existing string field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.219 void Wombat::MamaMsg::updateOpaque (const char * *name*, mama_fid_t *fid*, const void * *value*, size_t *size*)

Update the value of an existing opaque field.

If the field does not exist it is added.

Parameters:

name The name.

value The new value.

fid The field identifier.

size The size of the opaque in bytes

7.24.3.220 void Wombat::MamaMsg::updateOpaque (const MamaFieldDescriptor * *fieldDesc*, const void * *value*, size_t *size*)

Update the value of an existing opaque field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

size The size of the opaque in bytes

7.24.3.221 void Wombat::MamaMsg::updateDateTime (const char * *name*, mama_fid_t *fid*, const MamaDateTime & *value*)

Update a date/time field.

If the field does not exist it is added.

Parameters:

name The field name.

fid The field identifier.

value The value.

7.24.3.222 void Wombat::MamaMsg::updateDateTime (const MamaFieldDescriptor * *fieldDesc*, const MamaDateTime & *value*)

Update the value of an existing date/time field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.223 void Wombat::MamaMsg::updatePrice (const char * *name*, mama_fid_t *fid*, const MamaPrice & *value*)

Update a price field.

If the field does not exist it is added.

Parameters:

name The field name.

fid The field identifier.

value The value.

7.24.3.224 void Wombat::MamaMsg::updatePrice (const MamaFieldDescriptor * *fieldDesc*, const MamaPrice & *value*)

Update the value of an existing price field.

If the field does not exist it is added.

Parameters:

fieldDesc The field descriptor

value The new value.

7.24.3.225 mamaMsgType Wombat::MamaMsg::getType (void) const

Get the message type.

7.24.3.226 const char* Wombat::MamaMsg::getMsgTypeName (void) const

Get a human readable type name.

7.24.3.227 mamaMsgStatus Wombat::MamaMsg::getStatus (void) const

Get the msg status.

7.24.3.228 const char* Wombat::MamaMsg::getMsgStatusString (void) const

Get human readable message status.

7.24.3.229 void Wombat::MamaMsg::iterateFields (MamaMsgFieldIterator & *iterator*, const MamaDictionary * *dictionary*, void * *closure*) const

Iterate over all the fields.

7.24.3.230 const char* Wombat::MamaMsg::toString () const

Return a const char * representation the message.

The memory allocated by this method gets freed upon destroying the message or invoking toString() again.

Returns:

A string representation of the message.

7.24.3.231 void Wombat::MamaMsg::getFieldAsString (const char * *name*, mama_fid_t *fid*, char * *result*, size_t *maxResultLen*) const

Obtain a string representation the field with the given fid.

Parameters:

name The field name.

fid The field identifier.

result Buffer to put result.

maxResultLen Maximum size of buffer to put result.

7.24.3.232 void Wombat::MamaMsg::getFieldAsString (const MamaFieldDescriptor * *fieldDesc*, char * *result*, size_t *maxResultLen*) const

Obtain a string representation the field with the given fid.

Parameters:

fieldDesc The field descriptor

result Buffer to put result.

maxResultLen Maximum size of buffer to put result.

7.24.3.233 MamaMsgField* Wombat::MamaMsg::getField (const char * *name*, mama_fid_t *fid*) const

Obtain a the mamaMsgField with the given fid.

Parameters:

name The field name.

fid The field identifier.

7.24.3.234 MamaMsgField* Wombat::MamaMsg::getField (const MamaFieldDescriptor * *fieldDesc*) const

Obtain a the mamaMsgField with the given fid.

Parameters:

fieldDesc The field descriptor

7.24.3.235 bool Wombat::MamaMsg::tryField (const char * *name*, mama_fid_t *fid*) const

Test for the presence of the MamaMsgField with the given fid.

This method does not return the field.

Parameters:

name The field name.

fid The field identifier.

7.24.3.236 bool Wombat::MamaMsg::tryField (const MamaFieldDescriptor * *fieldDesc*) const

Test for the presence of the MamaMsgField with the given field descriptor.

This method does not return the field.

Parameters:

fieldDesc The field descriptor

7.24.3.237 bool Wombat::MamaMsg::tryField (const char * *name*, mama_fid_t *fid*, MamaMsgField * *result*) const

Try to obtain the MamaMsgField with the given fid.

Parameters:

name The field name.

fid The field identifier.

result The result (not modified if field not present)

7.24.3.238 bool Wombat::MamaMsg::tryField (const MamaFieldDescriptor * *fieldDesc*, MamaMsgField * *result*) const

Try to obtain the MamaMsgField with the given field descriptor.

Parameters:

fieldDesc The field descriptor

result The result (not modified if field not present)

7.24.3.239 bool Wombat::MamaMsg::tryFieldAsString (const char * *name*, mama_fid_t *fid*, char * *result*, size_t *maxResultLen*) const

Try to obtain a string representation the field with the given fid.

Parameters:

name The field name.

fid The field identifier.

result Buffer to put result.

maxResultLen Maximum size of buffer to put result.

Returns:

Whether the field was present.

7.24.3.240 bool Wombat::MamaMsg::tryFieldAsString (const MamaFieldDescriptor * *fieldDesc*, char * *result*, size_t *maxResultLen*) const

Try to obtain a string representation the field with the given fid.

Parameters:

fieldDesc The field descriptor

result Buffer to put result.

maxResultLen Maximum size of buffer to put result.

Returns:

Whether the field was present.

7.24.3.241 void Wombat::MamaMsg::getByteBuffer (const void *& *buffer*, size_t & *bufferLength*) const

Get the underlying message as an array of bytes.

The buffer still belongs to the underlying message so no attempt should be made to modify it.

Parameters:

buffer The byte array containing the buffer

bufferLength The length, in bytes of the returned buffer

7.24.3.242 void Wombat::MamaMsg::createForBridge (mamaBridge *bridgeImpl*)

Create a MamaMsg.

This will create a message using the native format for the bridge e.g. TIB/RV messages for the TIB/RV bridge. For middlewares which only do not have a native message format, a wombatmsg will be created.

Parameters:

bridgeImpl the bridge to use

7.24.3.243 MamaMsg* Wombat::MamaMsg::detach (void)

Normally the Mama API owns incoming mamaMsg objects and they go out of scope and are freed when the message callback returns.

Calling this method from the message callback creates a new C++ wrapper for the underlying message and transfers responsibility for calling destroy() to the caller.

The caller must also delete the pointer returned by detach(). Note that calling "delete msg.detach()" invokes destroy() so calling destroy is not necessary if the application calls delete.

7.24.3.244 bool Wombat::MamaMsg::isFromInbox (void) const

Whether this message is the result of a request needing a response.

7.24.3.245 mama_seqnum_t Wombat::MamaMsg::getSeqNum (void) const

7.24.3.246 bool Wombat::MamaMsg::getIsDefinitelyDuplicate (void) const

Return true if this message is definitely a duplicate message.

This condition will not occur with the current feed handlers.

7.24.3.247 bool Wombat::MamaMsg::getIsPossiblyDuplicate (void) const

Return true if this message is possibly a duplicate message.

This may occur in the event of a fault tolerant feed handler take over where the feed handler replays messages to prevent gaps.

7.24.3.248 bool Wombat::MamaMsg::getIsPossiblyDelayed (void) const

Return true if the message is possibly delayed.

This condition may be true during a fault-tolerant takeover.

7.24.3.249 bool Wombat::MamaMsg::getIsDefinitelyDelayed (void) const

Return true if the message is delayed.

This condition may be true during a fault-tolerant takeover.

7.24.3.250 bool Wombat::MamaMsg::getIsOutOfSequence (void) const

Return true when the FH sends the message out of sequence.

7.24.3.251 bool Wombat::MamaMsg::setNewBuffer (void * *buffer*, mama_size_t *size*)

Sets a new for an existing mamaMsg using the provided byte buffer.

The application is responsible for destroying the message.

Parameters:

buffer the new byte buffer

size size of buffer

Returns:

status

7.24.3.252 void* Wombat::MamaMsg::getNativeHandle (void) const

Return the native middleware message handle.

This is only intended for Wombat internal use.

7.24.3.253 void Wombat::MamaMsg::createFromMsg (mamaMsg *msg*, bool *destroyMsg* = `false`) const

Create the message from an existing mamaMsg.

7.24.3.254 void Wombat::MamaMsg::setMsg (mamaMsg *msg*)

Set the message to a different underlying C message.

Can be called multiple times on a single MamaMsg

7.24.3.255 const mamaMsg& Wombat::MamaMsg::getUnderlyingMsg (void) const

Return const reference to underlying mamaMsg.

7.24.3.256 mamaMsg Wombat::MamaMsg::getUnderlyingMsg (void)

Return the underlying mamaMsg (non const).

7.24.3.257 mamaPayloadType Wombat::MamaMsg::getPayloadType (void) const

Return the type of the payload message (wombat message or, if using a non-wombat message payload, RV or FAST message).

Returns:

payloadType The payload type.

7.24.3.258 void* Wombat::MamaMsg::getNativeMsg (void)

Get the native message structure for the underlying message.

Returns:

nativeMsg The resulting native handle.

7.24.3.259 MamaMsgField& Wombat::MamaMsg::begin (MamaMsgIterator & *theIterator*) const

Sets a iterator to be used with existing mamaMsg.

The iterator is set to the first MamaMsgField of the mamaMsg

Parameters:

theiterator iterator to be used

Returns:

first MamaMsgField

7.24.3.260 mamaMsgReply Wombat::MamaMsg::getReplyHandle (void) const

Get a copy of the reply Handle.

7.24.3.261 static void Wombat::MamaMsg::destroyReplyHandle (mamaMsgReply *replyHandle*) [static]

Destroy a copied reply Handle.

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

- MamaMsg.h

7.25 Wombat::MamaMsgField Class Reference

MamaMsg field representation.

`#include <MamaMsgField.h>`

Public Member Functions

- ~MamaMsgField ()
- MamaMsgField (void)
- MamaMsgField (mamaMsgField field)
- void clear ()

  Clear the field.

- void set (mamaMsgField field)

  Set this field to a different MAMA C API field.

- const MamaFieldDescriptor * getDescriptor () const
- mama_fid_t getFid () const

  Return the field identifier.

- const char * getName () const

  Return the field name.

- mamaFieldType getType () const

  Return the field type.

- const char * getTypeName () const

  Return the field type name.

- mama_bool_t getBool () const

  Get as a boolean field.

- char getChar () const

  Get as a character field.

- mama_i8_t getI8 () const

  Get as a I8 field.

- mama_u8_t getU8 () const

  Get as a U8 field.

- mama_i16_t getI16 () const
  Get as a I16 field.
- mama_u16_t getU16 () const
  Get as a U16 field.
- mama_i32_t getI32 () const
  Get as a I32 field.
- mama_u32_t getU32 () const
  Get as a U32 field.
- mama_i64_t getI64 () const
  Get as a I64 field.
- mama_u64_t getU64 () const
  Get as a U64 field.
- mama_f32_t getF32 () const
  Get as a f32 field.
- mama_f64_t getF64 () const
  Get as a f64 field.
- const char * getString () const
  Get as a const char field.*
- const void * getOpaque (mama_size_t &size) const
  Get as a const void field.*
- void getDateTime (MamaDateTime &result) const
  Get as a MamaDateTime field.
- void getPrice (MamaPrice &result) const
  Get as a MamaPrice field.
- void getMsg (MamaMsg &result) const
  Get as a MamaMsg field.
- void getVectorChar (const char *&result, mama_size_t &resultLen) const
  Get a vector of characters.

- void getVectorI8 (const mama_i8_t *&result, mama_size_t &resultLen) const

  Get a vector of signed 8-bit integers.

- void getVectorU8 (const mama_u8_t *&result, mama_size_t &resultLen) const

  Get a vector of unsigned 8-bit integers.

- void getVectorI16 (const mama_i16_t *&result, mama_size_t &resultLen) const

  Get a vector of signed 16-bit integers.

- void getVectorU16 (const mama_u16_t *&result, mama_size_t &resultLen) const

  Get a vector of unsigned 16-bit integers.

- void getVectorI32 (const mama_i32_t *&result, mama_size_t &resultLen) const

  Get a vector of signed 32-bit integers.

- void getVectorU32 (const mama_u32_t *&result, mama_size_t &resultLen) const

  Get a vector of unsigned 32-bit integers.

- void getVectorI64 (const mama_i64_t *&result, mama_size_t &resultLen) const

  Get a vector of signed 64-bit integers.

- void getVectorU64 (const mama_u64_t *&result, mama_size_t &resultLen) const

  Get a vector of unsigned 64-bit integers.

- void getVectorF32 (const mama_f32_t *&result, mama_size_t &resultLen) const

  Get a vector of 32-bit floats.

- void getVectorF64 (const mama_f64_t *&result, mama_size_t &resultLen) const

  Get a vector of 64-bit floats.

- void getVectorString (const char **&result, mama_size_t &resultLen) const

  Get a vector of strings.

- void getVectorMsg (const MamaMsg **&result, mama_size_t &resultLen) const

  Get a vector of submessages field.

- void getVectorMsgDetached (const MamaMsg **&result, mama_size_t &resultLen) const

  Get a vector of submessages field.

- void getAsString (char *result, mama_size_t maxResultLen) const

  Return a string representation the field with the given fid.

- void updateBool (mama_bool_t value)

  Update the specified field with a new bool value.

- void updateChar (char value)

  Update the specified field with a new char value.

- void updateI8 (mama_i8_t value)

  Update the specified field with a new i8 value.

- void updateU8 (mama_u8_t value)

  Update the specified field with a new u8 value.

- void updateI16 (mama_i16_t value)

  Update the specified field with a new i16 value.

- void updateU16 (mama_u16_t value)

  Update the specified field with a new u16 value.

- void updateI32 (mama_i32_t value)

  Update the specified field with a new i32 value.

- void updateU32 (mama_u32_t value)

  Update the specified field with a new u32 value.

- void updateI64 (mama_i64_t value)

  Update the specified field with a new i64 value.

- void updateU64 (mama_u64_t value)

  Update the specified field with a new u64 value.

- void updateF32 (mama_f32_t value)

  Update the specified field with a new f32 value.

- void updateF64 (mama_f64_t value)

  Update the specified field with a new f64 value.

- void updateDateTime (const mamaDateTime value)

  Update the specified field with a new date/time value.

- void updateDateTime (const MamaDateTime value)

  Update the specified field with a new date/time value.

- void updatePrice (const mamaPrice value)

  Update the specified field with a new price value.

- void updatePrice (const MamaPrice value)

  Update the specified field with a new price value.

- bool operator== (const MamaMsgField &) const
- bool operator!= (const MamaMsgField &) const

7.25.1 Detailed Description

MamaMsg field representation.

7.25.2 Constructor & Destructor Documentation

7.25.2.1 Wombat::MamaMsgField::~MamaMsgField ()

7.25.2.2 Wombat::MamaMsgField::MamaMsgField (void)

7.25.2.3 Wombat::MamaMsgField::MamaMsgField (mamaMsgField *field*)

7.25.3 Member Function Documentation

7.25.3.1 void Wombat::MamaMsgField::clear ()

Clear the field.

7.25.3.2 void Wombat::MamaMsgField::set (mamaMsgField *field*)

Set this field to a different MAMA C API field.

7.25.3.3 const MamaFieldDescriptor* Wombat::MamaMsgField::getDescriptor () const

7.25.3.4 mama_fid_t Wombat::MamaMsgField::getFid () const

Return the field identifier.

Returns:

The fid.

7.25.3.5 const char* Wombat::MamaMsgField::getName () const

Return the field name.

Returns:

The name.

7.25.3.6 mamaFieldType Wombat::MamaMsgField::getType () const

Return the field type.

Returns:

The type.

7.25.3.7 const char* Wombat::MamaMsgField::getTypeName () const

Return the field type name.

The type name is a human readable representation of the type.

Returns:

The type name.

7.25.3.8 mama_bool_t Wombat::MamaMsgField::getBool () const

Get as a boolean field.

Returns:

The boolean value.

7.25.3.9 char Wombat::MamaMsgField::getChar () const

Get as a character field.

Returns:

The character value.

7.25.3.10 mama_i8_t Wombat::MamaMsgField::getI8 () const

Get as a I8 field.

Returns:

The integer value.

7.25.3.11 mama_u8_t Wombat::MamaMsgField::getU8 () const

Get as a U8 field.

Returns:

The integer value.

7.25.3.12 mama_i16_t Wombat::MamaMsgField::getI16 () const

Get as a I16 field.

Returns:

The integer value.

7.25.3.13 mama_u16_t Wombat::MamaMsgField::getU16 () const

Get as a U16 field.

Returns:

The integer value.

7.25.3.14 mama_i32_t Wombat::MamaMsgField::getI32 () const

Get as a I32 field.

Returns:

The integer value.

7.25.3.15 mama_u32_t Wombat::MamaMsgField::getU32 () const

Get as a U32 field.

Returns:

The integer value.

7.25.3.16 mama_i64_t Wombat::MamaMsgField::getI64 () const

Get as a I64 field.

Returns:

The field value as a long.

7.25.3.17 mama_u64_t Wombat::MamaMsgField::getU64 () const

Get as a U64 field.

Returns:

The field value as a long.

7.25.3.18 mama_f32_t Wombat::MamaMsgField::getF32 () const

Get as a f32 field.

Returns:

The field value as a 32 bit floating point number.

7.25.3.19 mama_f64_t Wombat::MamaMsgField::getF64 () const

Get as a f64 field.

Returns:

The field value as a 64 bit floating point number.

7.25.3.20 const char* Wombat::MamaMsgField::getString () const

Get as a const char* field.

Returns:

the string value.

7.25.3.21 const void* Wombat::MamaMsgField::getOpaque (mama_size_t & *size*) const

Get as a const void* field.

Returns:

The opaque value.

7.25.3.22 void Wombat::MamaMsgField::getDateTime (MamaDateTime & *result*) const

Get as a MamaDateTime field.

Parameters:

result The date/time value.

7.25.3.23 void Wombat::MamaMsgField::getPrice (MamaPrice & *result*) const

Get as a MamaPrice field.

Parameters:

result The price value.

7.25.3.24 void Wombat::MamaMsgField::getMsg (MamaMsg & *result*) const

Get as a MamaMsg field.

Parameters:

result The msg value.

7.25.3.25 void Wombat::MamaMsgField::getVectorChar (const char *& *result*, mama_size_t & *resultLen*) const

Get a vector of characters.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.26 void Wombat::MamaMsgField::getVectorI8 (const mama_i8_t *& *result*, mama_size_t & *resultLen*) const

Get a vector of signed 8-bit integers.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.27 void Wombat::MamaMsgField::getVectorU8 (const mama_u8_t *& *result*, mama_size_t & *resultLen*) const

Get a vector of unsigned 8-bit integers.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.28 void Wombat::MamaMsgField::getVectorI16 (const mama_i16_t *& *result*, mama_size_t & *resultLen*) const

Get a vector of signed 16-bit integers.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.29 void Wombat::MamaMsgField::getVectorU16 (const mama_u16_t *& *result*, mama_size_t & *resultLen*) const

Get a vector of unsigned 16-bit integers.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.30 void Wombat::MamaMsgField::getVectorI32 (const mama_i32_t *& *result*, mama_size_t & *resultLen*) const

Get a vector of signed 32-bit integers.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.31 void Wombat::MamaMsgField::getVectorU32 (const mama_u32_t *& *result*, mama_size_t & *resultLen*) const

Get a vector of unsigned 32-bit integers.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.32 void Wombat::MamaMsgField::getVectorI64 (const mama_i64_t *& *result*, mama_size_t & *resultLen*) const

Get a vector of signed 64-bit integers.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.33 void Wombat::MamaMsgField::getVectorU64 (const mama_u64_t *& *result*, mama_size_t & *resultLen*) const

Get a vector of unsigned 64-bit integers.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.34 void Wombat::MamaMsgField::getVectorF32 (const mama_f32_t *& *result*, mama_size_t & *resultLen*) const

Get a vector of 32-bit floats.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.35 void Wombat::MamaMsgField::getVectorF64 (const mama_f64_t *& *result*, mama_size_t & *resultLen*) const

Get a vector of 64-bit floats.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.36 void Wombat::MamaMsgField::getVectorString (const char **& *result*, mama_size_t & *resultLen*) const

Get a vector of strings.

Parameters:

result (out) the vector.

resultLen (out) the size of the vector.

7.25.3.37 void Wombat::MamaMsgField::getVectorMsg (const MamaMsg **& *result*, mama_size_t & *resultLen*) const

Get a vector of submessages field.

Note: The vector is only valid for the lifetime of the field object which, when iterating over fields in a message is only as long as the individual callback itself. Use getVectorMsgDetached if it is necessary to keep the vector of messages longer than the lifetime of the field.

Parameters:

result (out) vector.

resultLen (out) the size of the vector.

7.25.3.38 void Wombat::MamaMsgField::getVectorMsgDetached (const MamaMsg **& *result*, mama_size_t & *resultLen*) const

Get a vector of submessages field.

Deallocating the memory allocated for array and it members will become the responsibility of the caller.

Parameters:

result (out) vector.

resultLen (out) the size of the vector.

7.25.3.39 void Wombat::MamaMsgField::getAsString (char * *result*, mama_size_t *maxResultLen*) const

Return a string representation the field with the given fid.

Parameters:

result Buffer to put result.

maxResultLen Maximum size of buffer to put result.

7.25.3.40 void Wombat::MamaMsgField::updateBool (mama_bool_t *value*)

Update the specified field with a new bool value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.41 void Wombat::MamaMsgField::updateChar (char *value*)

Update the specified field with a new char value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.42 void Wombat::MamaMsgField::updateI8 (mama_i8_t *value*)

Update the specified field with a new i8 value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_-STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.43 void Wombat::MamaMsgField::updateU8 (mama_u8_t *value*)

Update the specified field with a new u8 value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_-STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.44 void Wombat::MamaMsgField::updateI16 (mama_i16_t *value*)

Update the specified field with a new i16 value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_-STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.45 void Wombat::MamaMsgField::updateU16 (mama_u16_t *value*)

Update the specified field with a new u16 value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.46 void Wombat::MamaMsgField::updateI32 (mama_i32_t *value*)

Update the specified field with a new i32 value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.47 void Wombat::MamaMsgField::updateU32 (mama_u32_t *value*)

Update the specified field with a new u32 value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.48 void Wombat::MamaMsgField::updateI64 (mama_i64_t *value*)

Update the specified field with a new i64 value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_-STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.49 void Wombat::MamaMsgField::updateU64 (mama_u64_t *value*)

Update the specified field with a new u64 value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_-STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.50 void Wombat::MamaMsgField::updateF32 (mama_f32_t *value*)

Update the specified field with a new f32 value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_-STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.51 void Wombat::MamaMsgField::updateF64 (mama_f64_t *value*)

Update the specified field with a new f64 value.

Parameters:

value The new value for the field.

Exceptions:

MamaStatus exception with the following possible status values. MAMA_STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.52 void Wombat::MamaMsgField::updateDateTime (const mamaDateTime *value*)

Update the specified field with a new date/time value.

Parameters:

value The new value for the field (mamaDateTime object).

Exceptions:

MamaStatus exception with the following possible status values. MAMA_STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.53 void Wombat::MamaMsgField::updateDateTime (const MamaDateTime *value*)

Update the specified field with a new date/time value.

Parameters:

value The new value for the field (MamaDateTime object).

Exceptions:

MamaStatus exception with the following possible status values. MAMA_STATUS_WRONG_FIELD_TYPE The existing field type does not match

the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.54 void Wombat::MamaMsgField::updatePrice (const mamaPrice *value*)

Update the specified field with a new price value.

Parameters:

value The new value for the field (mamaPrice object).

Exceptions:

MamaStatus exception with the following possible status values. MAMA_-STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.55 void Wombat::MamaMsgField::updatePrice (const MamaPrice *value*)

Update the specified field with a new price value.

Parameters:

value The new value for the field (MamaPrice object).

Exceptions:

MamaStatus exception with the following possible status values. MAMA_-STATUS_WRONG_FIELD_TYPE The existing field type does not match the type of the update method called. MAMA_STATUS_NULL_ARG The field passed to the C function is NULL. MAMA_STATUS_INVALID_ARG The underlying bridge field is NULL.

7.25.3.56 bool Wombat::MamaMsgField::operator== (const MamaMsgField &) const

7.25.3.57 bool Wombat::MamaMsgField::operator!= (const MamaMsgField &) const

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

- MamaMsgField.h

7.26 Wombat::MamaMsgFieldIterator Class Reference

Callback class for iterating over all fields in a message.

`#include <MamaMsgFieldIterator.h>`

Public Member Functions

- virtual ~MamaMsgFieldIterator ()
- virtual void onField (const MamaMsg &msg, const MamaMsgField &field, void *closure)=0

7.26.1 Detailed Description

Callback class for iterating over all fields in a message.

This is necessary because some messaging implementations do not provide efficient indexed access to fields.

7.26.2 Constructor & Destructor Documentation

7.26.2.1 virtual Wombat::MamaMsgFieldIterator::~MamaMsgFieldIterator () [virtual]

```
38         {
39         };
```

7.26.3 Member Function Documentation

7.26.3.1 virtual void Wombat::MamaMsgFieldIterator::onField (const MamaMsg & *msg*, const MamaMsgField & *field*, void * *closure*) [pure virtual]

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

- MamaMsgFieldIterator.h

7.27 Wombat::MamaMsgIterator Class Reference

Iterator class for mamaMsg.

`#include <MamaMsg.h>`

Public Member Functions

- MamaMsgIterator (const MamaDictionary *dictionary)
- MamaMsgIterator ()
- MamaMsgIterator (const MamaMsgIterator &copy)
- ~MamaMsgIterator ()
- void SetDict (const MamaDictionary *dictionary)
- MamaMsgIterator & operator= (const MamaMsgIterator &)
- bool operator== (const MamaMsgIterator &) const
- bool operator!= (const MamaMsgIterator &) const
- MamaMsgField & operator * ()
- MamaMsgField * operator → ()
- MamaMsgIterator & operator++ ()

Protected Attributes

- mamaMsgIterator myImpl
- MamaMsgField mMsgField

Friends

- class MamaMsg

7.27.1 Detailed Description

Iterator class for mamaMsg.

Once an iterator has been created it can be set to the beginning of a mamaMsg and used to iterate over the MamaMsgFields.

An iterator can only be used with one message at a time, and only one iterator can be on one message at a time

Only forward iterating is supported

The dictionary to be used with the fields can be set in the iterator

7.27.2 Constructor & Destructor Documentation

7.27.2.1 Wombat::MamaMsgIterator::MamaMsgIterator (const MamaDictionary * *dictionary*)

7.27.2.2 Wombat::MamaMsgIterator::MamaMsgIterator ()

7.27.2.3 Wombat::MamaMsgIterator::MamaMsgIterator (const MamaMsgIterator & *copy*)

7.27.2.4 Wombat::MamaMsgIterator::~MamaMsgIterator ()

7.27.3 Member Function Documentation

7.27.3.1 void Wombat::MamaMsgIterator::SetDict (const MamaDictionary * *dictionary*)

7.27.3.2 MamaMsgIterator& Wombat::MamaMsgIterator::operator= (const MamaMsgIterator &)

7.27.3.3 bool Wombat::MamaMsgIterator::operator== (const MamaMsgIterator &) const

7.27.3.4 bool Wombat::MamaMsgIterator::operator!= (const MamaMsgIterator &) const

7.27.3.5 MamaMsgField& Wombat::MamaMsgIterator::operator * ()

7.27.3.6 MamaMsgField* Wombat::MamaMsgIterator::operator -> ()

7.27.3.7 MamaMsgIterator& Wombat::MamaMsgIterator::operator++ ()

7.27.4 Friends And Related Function Documentation

7.27.4.1 friend class MamaMsg [friend]

7.27.5 Member Data Documentation

7.27.5.1 mamaMsgIterator Wombat::MamaMsgIterator::myImpl [protected]

7.27.5.2 MamaMsgField Wombat::MamaMsgIterator::mMsgField [protected]

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

- MamaMsg.h

7.28 Wombat::MamaMsgQual Class Reference

The MamaMsgQual class is a wrapper/utility class which provides useful interrogation, comparison and manipulation facilities for the Mama Message Qualifier data field (w-MsgQual) which is a "bit-map" used to convey duplicate, delayed and out-of-sequence information about messages.

`#include <MamaMsgQual.h>`

Public Member Functions

- MamaMsgQual ()
- MamaMsgQual (mama_u16_t value)
- MamaMsgQual (const MamaMsgQual &copy)
- ~MamaMsgQual ()
- MamaMsgQual & operator= (const MamaMsgQual &rhs)
- bool operator== (const MamaMsgQual &rhs) const
- bool operator== (mama_u16_t rhs) const
- bool operator!= (const MamaMsgQual &rhs) const
- bool operator!= (mama_u16_t rhs) const
- void clear ()
- void setValue (mama_u16_t value)
- void setIsDefinatelyDuplicate (bool tf)
- void setIsPossiblyDuplicate (bool tf)
- void setIsDefinatelyDelayed (bool tf)
- void setIsPossiblyDelayed (bool tf)
- void setIsOutOfSequence (bool tf)
- mama_u16_t getValue () const
- bool getIsDefinatelyDuplicate () const
- bool getIsPossiblyDuplicate () const
- bool getIsDefinatelyDelayed () const
- bool getIsPossiblyDelayed () const
- bool getIsOutOfSequence () const
- void getAsString (char *result, mama_size_t maxLen) const
- const char * getAsString () const

Return a string representation of the message qualifier.

Static Public Member Functions

- static void getAsString (const mama_u16_t &value, char *result, mama_size_t maxLen)

  Static helper function to convert from the raw 16bit integer representation directly to a colon delimited string of conditions.

7.28.1 Detailed Description

The MamaMsgQual class is a wrapper/utility class which provides useful interrogation, comparison and manipulation facilities for the Mama Message Qualifier data field (w-MsgQual) which is a "bit-map" used to convey duplicate, delayed and out-of-sequence information about messages.

7.28.2 Constructor & Destructor Documentation

7.28.2.1 Wombat::MamaMsgQual::MamaMsgQual ()

7.28.2.2 Wombat::MamaMsgQual::MamaMsgQual (mama_u16_t *value*)

7.28.2.3 Wombat::MamaMsgQual::MamaMsgQual (const MamaMsgQual & *copy*)

7.28.2.4 Wombat::MamaMsgQual::~MamaMsgQual ()

7.28.3 Member Function Documentation

7.28.3.1 MamaMsgQual& Wombat::MamaMsgQual::operator= (const MamaMsgQual & *rhs*)

7.28.3.2 bool Wombat::MamaMsgQual::operator== (const MamaMsgQual & *rhs*) const

7.28.3.3 bool Wombat::MamaMsgQual::operator== (mama_u16_t *rhs*) const

7.28.3.4 bool Wombat::MamaMsgQual::operator!= (const MamaMsgQual & *rhs*) const

```
51          {
52              return ! operator== (rhs);
53          }
```

7.28.3.5 bool Wombat::MamaMsgQual::operator!= (mama_u16_t *rhs*) const

```
55          {
56              return ! operator== (rhs);
57          }
```

7.28.3.6 void Wombat::MamaMsgQual::clear ()

7.28.3.7 void Wombat::MamaMsgQual::setValue (mama_u16_t *value*)

7.28.3.8 void Wombat::MamaMsgQual::setIsDefinatelyDuplicate (bool *tf*)

7.28.3.9 void Wombat::MamaMsgQual::setIsPossiblyDuplicate (bool *tf*)

7.28.3.10 void Wombat::MamaMsgQual::setIsDefinatelyDelayed (bool *tf*)

7.28.3.11 void Wombat::MamaMsgQual::setIsPossiblyDelayed (bool *tf*)

7.28.3.12 void Wombat::MamaMsgQual::setIsOutOfSequence (bool *tf*)

7.28.3.13 mama_u16_t Wombat::MamaMsgQual::getValue () const

7.28.3.14 bool Wombat::MamaMsgQual::getIsDefinatelyDuplicate () const

7.28.3.15 bool Wombat::MamaMsgQual::getIsPossiblyDuplicate () const

7.28.3.16 bool Wombat::MamaMsgQual::getIsDefinatelyDelayed () const

7.28.3.17 bool Wombat::MamaMsgQual::getIsPossiblyDelayed () const

7.28.3.18 bool Wombat::MamaMsgQual::getIsOutOfSequence () const

7.28.3.19 void Wombat::MamaMsgQual::getAsString (char * *result*, mama_size_t *maxLen*) const

7.28.3.20 const char* Wombat::MamaMsgQual::getAsString () const

Return a string representation of the message qualifier.

Note that the alternative getAsString() method is more efficient because this method must allocate a temporary buffer (automatically destroyed upon object destruction).

7.28.3.21 static void Wombat::MamaMsgQual::getAsString (const mama_u16_t & *value*, char * *result*, mama_size_t *maxLen*) [static]

Static helper function to convert from the raw 16bit integer representation directly to a colon delimited string of conditions.

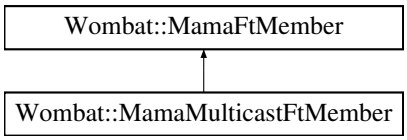
The documentation for this class was generated from the following file:

- MamaMsgQual.h

7.29 Wombat::MamaMulticastFtMember Class Reference

`#include <MamaFt.h>`

Inheritance diagram for Wombat::MamaMulticastFtMember::



Public Member Functions

- void setup (MamaQueue *queue, MamaFtMemberCallback *handler, MamaTransport *transport, const char *groupName, mama_u32_t weight, mama_f64_t heartbeatInterval, mama_f64_t timeoutInterval, void *closure=NULL)

7.29.1 Member Function Documentation

7.29.1.1 void Wombat::MamaMulticastFtMember::setup (MamaQueue * *queue*, MamaFtMemberCallback * *handler*, MamaTransport * *transport*, const char * *groupName*, mama_u32_t *weight*, mama_f64_t *heartbeatInterval*, mama_f64_t *timeoutInterval*, void * *closure* = NULL) [virtual]

Implements Wombat::MamaFtMember.

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

- MamaFt.h

7.30 Wombat::MamaPrice Class Reference

MamaPrice is a special data type for representing floating point numbers that often require special formatting for display purposes, such as prices.

`#include <MamaPrice.h>`

Public Member Functions

- MamaPrice ()
- MamaPrice (double value, mamaPricePrecision precision=MAMA_PRICE_PREC_100)
- MamaPrice (const MamaPrice &copy)
- ~MamaPrice ()
- MamaPrice & operator= (const MamaPrice &rhs)
- MamaPrice & operator+= (const MamaPrice &rhs)
- MamaPrice & operator-= (const MamaPrice &rhs)
- bool operator== (const MamaPrice &rhs) const
- bool operator== (double rhs) const
- bool operator!= (const MamaPrice &rhs) const
- bool operator!= (double rhs) const
- bool operator< (const MamaPrice &rhs) const
- bool operator< (double rhs) const
- bool operator> (const MamaPrice &rhs) const
- bool operator> (double rhs) const
- MamaPrice operator- (const MamaPrice &rhs) const

  Subtraction operator.

- MamaPrice operator- (double rhs) const

  Subtraction operator for double.

- MamaPrice operator- () const

  Negation operator.

- MamaPrice operator+ (const MamaPrice &rhs) const

  Addition operator.

- MamaPrice operator+ (double rhs) const

  Addition operator for double.

- double compare (const MamaPrice &rhs) const
- void clear ()
- void set (double priceValue, mamaPriceHints hints=0)

- void setValue (double value)
- void setPrecision (mamaPricePrecision precision)
- void setHints (mamaPriceHints hints)
- void setIsValidPrice (bool valid)
- double getValue () const
- mamaPricePrecision getPrecision () const
- mamaPriceHints getHints () const
- bool getIsValidPrice () const
- void setFromString (const char *str)
- void getAsString (char *result, mama_size_t maxLen) const
- const char * getAsString () const

  Return a string representation of the price.

- void negate ()

  Negate the price value.

- bool isZero () const

  Return whether the price has a value equivalent to zero.

- mamaPrice getCValue ()
- const mamaPrice getCValue () const

Static Public Member Functions

- static mamaPricePrecision decimals2Precision (mama_i32_t places)

  Return the appropriate precision code for a given number of decimal places.

- static mamaPricePrecision denom2Precision (mama_i32_t denominator)

  Return the appropriate precision code for a given fractional denominator.

- static mama_i32_t precision2Decimals (mamaPricePrecision precision)

  Return the number of decimal places for a given precision code.

- static mama_i32_t precision2Denom (mamaPricePrecision precision)

  Return the fractional denominator for a given precision code.

7.30.1 Detailed Description

MamaPrice is a special data type for representing floating point numbers that often require special formatting for display purposes, such as prices.

MamaPrice contains the 64-bit (double precision) floating point value and an optional display hint. The set of display hints includes hints for:

- a number of decimal places,
- a fractional denominator that are powers of two, and
- hints for special denominators used in the finance industry (e.g., halves of 32nds).

7.30.2 Constructor & Destructor Documentation

7.30.2.1 Wombat::MamaPrice::MamaPrice ()

7.30.2.2 Wombat::MamaPrice::MamaPrice (double *value*, mamaPricePrecision *precision* = MAMA_PRICE_PREC_100)

7.30.2.3 Wombat::MamaPrice::MamaPrice (const MamaPrice & *copy*)

7.30.2.4 Wombat::MamaPrice::~MamaPrice ()

7.30.3 Member Function Documentation

7.30.3.1 MamaPrice& Wombat::MamaPrice::operator= (const MamaPrice & *rhs*)

7.30.3.2 MamaPrice& Wombat::MamaPrice::operator+= (const MamaPrice & *rhs*)

7.30.3.3 MamaPrice& Wombat::MamaPrice::operator-= (const MamaPrice & *rhs*)

7.30.3.4 bool Wombat::MamaPrice::operator== (const MamaPrice & *rhs*) const

7.30.3.5 bool Wombat::MamaPrice::operator== (double *rhs*) const

7.30.3.6 bool Wombat::MamaPrice::operator!= (const MamaPrice & *rhs*) const

```
60          {
61              return ! operator== (rhs);
62          }
```

7.30.3.7 bool Wombat::MamaPrice::operator!= (double *rhs*) const

```
64          {
65              return ! operator== (rhs);
66          }
```

7.30.3.8 bool Wombat::MamaPrice::operator< (const MamaPrice & *rhs*) const

7.30.3.9 bool Wombat::MamaPrice::operator< (double *rhs*) const

7.30.3.10 bool Wombat::MamaPrice::operator> (const MamaPrice & *rhs*) const

7.30.3.11 bool Wombat::MamaPrice::operator> (double *rhs*) const

7.30.3.12 MamaPrice Wombat::MamaPrice::operator- (const MamaPrice & *rhs*) const

Subtraction operator.

Note: this operator creates a temporary object.

```
77          {
78              return MamaPrice (getValue() - rhs.getValue(),
79                                rhs.getPrecision ());
80          }
```

7.30.3.13 MamaPrice Wombat::MamaPrice::operator- (double *rhs*) const

Subtraction operator for double.

Note: this operator creates a temporary object.

```
87              {
88              return MamaPrice (getValue     () - rhs,
89                                getPrecision ());
90          }
```

7.30.3.14 MamaPrice Wombat::MamaPrice::operator- () const

Negation operator.

Note: this operator creates a temporary object.

```
97          {
98              return MamaPrice (-getValue    (),
99                                getPrecision ());
100          }
```

7.30.3.15 MamaPrice Wombat::MamaPrice::operator+ (const MamaPrice & *rhs*) const

Addition operator.

Note: this operator creates a temporary object.

```
106         {
107             return MamaPrice (getValue         () + rhs.getValue (),
108                               rhs.getPrecision ());
109         }
```

7.30.3.16 MamaPrice Wombat::MamaPrice::operator+ (double *rhs*) const

Addition operator for double.

Note: this operator creates a temporary object.

```
116         {
117             return MamaPrice (getValue     () + rhs,
118                               getPrecision ());
119         }
```

7.30.3.17 double Wombat::MamaPrice::compare (const MamaPrice & *rhs*) const

7.30.3.18 void Wombat::MamaPrice::clear ()

7.30.3.19 void Wombat::MamaPrice::set (double *priceValue*, mamaPriceHints *hints* = 0)

7.30.3.20 void Wombat::MamaPrice::setValue (double *value*)

7.30.3.21 void Wombat::MamaPrice::setPrecision (mamaPricePrecision *precision*)

7.30.3.22 void Wombat::MamaPrice::setHints (mamaPriceHints *hints*)

7.30.3.23 void Wombat::MamaPrice::setIsValidPrice (bool *valid*)

7.30.3.24 double Wombat::MamaPrice::getValue () const

7.30.3.25 mamaPricePrecision Wombat::MamaPrice::getPrecision () const

7.30.3.26 mamaPriceHints Wombat::MamaPrice::getHints () const

7.30.3.27 bool Wombat::MamaPrice::getIsValidPrice () const

7.30.3.28 void Wombat::MamaPrice::setFromString (const char * *str*)

7.30.3.29 void Wombat::MamaPrice::getAsString (char * *result*, mama_size_t *maxLen*) const

7.30.3.30 const char* Wombat::MamaPrice::getAsString () const

Return a string representation of the price.

Note that the alternative getAsString() method is more efficient because this method must allocate a temporary buffer (automatically destroyed upon object destruction).

7.30.3.31 void Wombat::MamaPrice::negate ()

Negate the price value.

Hints and precisions are not affected.

7.30.3.32 bool Wombat::MamaPrice::isZero () const

Return whether the price has a value equivalent to zero.

It may not be exactly 0.0, but we check against +/- epsilon.

7.30.3.33 static mamaPricePrecision Wombat::MamaPrice::decimals2Precision (mama_i32_t *places*) [static]

Return the appropriate precision code for a given number of decimal places.

7.30.3.34 static mamaPricePrecision Wombat::MamaPrice::denom2Precision (mama_i32_t *denominator*) [static]

Return the appropriate precision code for a given fractional denominator.

7.30.3.35 static mama_i32_t Wombat::MamaPrice::precision2Decimals (mamaPricePrecision *precision*) [static]

Return the number of decimal places for a given precision code.

7.30.3.36 static mama_i32_t Wombat::MamaPrice::precision2Denom (mamaPricePrecision *precision*) [static]

Return the fractional denominator for a given precision code.

7.30.3.37 mamaPrice Wombat::MamaPrice::getCValue ()

7.30.3.38 const mamaPrice Wombat::MamaPrice::getCValue () const

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

- MamaPrice.h

7.31 Wombat::MamaPublisher Class Reference

The publisher class publishes messages to basic or market data subscriptions depending on the underlying transport.

`#include <MamaPublisher.h>`

Public Member Functions

- virtual ~MamaPublisher (void)
- MamaPublisher (void)
- virtual void create (MamaTransport *transport, const char *topic, const char *source=NULL, const char *root=NULL)

  Create a MAMA publisher for the corresponding transport.

- virtual void send (MamaMsg *msg) const
- virtual void sendWithThrottle (MamaMsg *msg, MamaSendCompleteCallback *cb, void *closure) const
- virtual void sendFromInbox (MamaInbox *inbox, MamaMsg *msg) const
- virtual void sendFromInboxWithThrottle (MamaInbox *inbox, MamaMsg *msg, MamaSendCompleteCallback *cb, void *closure) const
- virtual void sendReplyToInbox (const MamaMsg &request, MamaMsg *reply) const
- virtual void sendReplyToInbox (mamaMsgReply replyHandle, MamaMsg *reply) const
- virtual void destroy (void)

Protected Member Functions

- MamaPublisher (MamaPublisherImpl *)

Protected Attributes

- MamaPublisherImpl * mPimpl

7.31.1 Detailed Description

The publisher class publishes messages to basic or market data subscriptions depending on the underlying transport.

For basic transports the source parameter must be NULL.

7.31.2 Constructor & Destructor Documentation

7.31.2.1 virtual Wombat::MamaPublisher::~MamaPublisher (void) [virtual]

7.31.2.2 Wombat::MamaPublisher::MamaPublisher (void)

7.31.2.3 Wombat::MamaPublisher::MamaPublisher (MamaPublisherImpl *) [protected]

7.31.3 Member Function Documentation

7.31.3.1 virtual void Wombat::MamaPublisher::create (MamaTransport * *transport*, const char * *topic*, const char * *source* = NULL, const char * *root* = NULL) [virtual]

Create a MAMA publisher for the corresponding transport.

If the transport is a market data transport, as opposed to a "basic" transport, the topic corresponds to the symbol. For a basic transport, the source and root get ignored.

Parameters:

transport The transport to use. Must be a basic transport.

topic for basic publishers. Symbol for market data topics.

source The source for market data publishers.

root The root for market data publishers. Used internally.

7.31.3.2 virtual void Wombat::MamaPublisher::send (MamaMsg * *msg*) const [virtual]

7.31.3.3 virtual void Wombat::MamaPublisher::sendWithThrottle (MamaMsg * *msg*, MamaSendCompleteCallback * *cb*, void * *closure*) const [virtual]

7.31.3.4 virtual void Wombat::MamaPublisher::sendFromInbox (MamaInbox * *inbox*, MamaMsg * *msg*) const [virtual]

7.31.3.5 virtual void Wombat::MamaPublisher::sendFromInboxWithThrottle (MamaInbox * *inbox*, MamaMsg * *msg*, MamaSendCompleteCallback * *cb*, void * *closure*) const [virtual]

7.31.3.6 virtual void Wombat::MamaPublisher::sendReplyToInbox (const MamaMsg & *request*, MamaMsg * *reply*) const [virtual]

7.31.3.7 virtual void Wombat::MamaPublisher::sendReplyToInbox (mamaMsgReply *replyHandle*, MamaMsg * *reply*) const [virtual]

7.31.3.8 virtual void Wombat::MamaPublisher::destroy (void) [virtual]

7.31.4 Member Data Documentation

7.31.4.1 MamaPublisherImpl* Wombat::MamaPublisher::mPimpl [protected]

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

- MamaPublisher.h

7.32 Wombat::MamaPublishTopic Class Reference

`#include <MamaDQPublisherManager.h>`

Public Attributes

- const char * mSymbol
- MamaDQPublisher * mPub
- void * mCache

Protected Member Functions

- MamaPublishTopic ()
- void set (mamaPublishTopic *pubInfo)

Friends

- struct MamaDQPublisherManagerImpl

7.32.1 Constructor & Destructor Documentation

7.32.1.1 Wombat::MamaPublishTopic::MamaPublishTopic () [protected]

```
46 {};
```

7.32.2 Member Function Documentation

7.32.2.1 void Wombat::MamaPublishTopic::set (mamaPublishTopic * *pubInfo*) [protected]

7.32.3 Friends And Related Function Documentation

7.32.3.1 friend struct MamaDQPublisherManagerImpl [friend]

7.32.4 Member Data Documentation

7.32.4.1 const char* Wombat::MamaPublishTopic::mSymbol

7.32.4.2 MamaDQPublisher* Wombat::MamaPublishTopic::mPub

7.32.4.3 void* Wombat::MamaPublishTopic::mCache

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

- MamaDQPublisherManager.h

7.33 Wombat::MamaQueue Class Reference

Queue allows applications to dispatch events in order with multiple threads using a single MamaDispatcher for each queue.

`#include <MamaQueue.h>`

Public Member Functions

- MamaQueue (void)
- MamaQueue (mamaQueue cQueue)
- virtual ~MamaQueue (void)
- virtual void create (mamaBridge bridgeImpl)

  Create a queue.

- virtual void create (mamaBridge bridgeImpl, void *nativeQueue)
- virtual void dispatch ()

  Dispatch message.

- virtual void timedDispatch (uint64_t timeout)

  Dispatch messages until timeout (see release notes for details).

- virtual void dispatchEvent ()

  Dispatch a single event from the specified queue.

- virtual void enqueueEvent (MamaQueueEventCallback *callback, void *closure)

  Add a user event to a queue.

- virtual void enqueueEvent (MamaQueueEventCallback &callback, void *closure)
- virtual void stopDispatch ()

  stopDispatch the queue.

- virtual size_t getEventCount ()

  Returns the number of events currently on the queue.

- virtual void setEnqueueCallback (MamaQueueEnqueueCallback *cb, void *closure)

  Set a callback which will be invoked as each event is added to the underlying event queue.

- virtual void setQueueMonitorCallback (MamaQueueMonitorCallback *cb, void *closure)

  Register an object to receive callbacks for monitoring the behaviour of the MamaQueue.

- virtual void setHighWatermark (size_t highWatermark)

  Specify a high watermark for events on the queue.

- virtual size_t getHighWatermark (void) const

  Return the high water mark as set via `setHighWaterMark()`.

- virtual void setLowWatermark (size_t lowWatermark)

  Set the low watermark.

- virtual size_t getLowWatermark (void) const

  Return the low water mark as set via `setLowWaterMark()`.

- virtual void setQueueName (const char *name)

  Associate a name identifier with the event queue.

- virtual const char * getQueueName () const

  Retrieve the string name identifier for the queue as specified from a call to `setQueueName()`.

- virtual const char * getQueueBridgeName () const

  Retrieve the string name identifier for the queue's bridge.

- virtual void destroy ()

  Destroy a queue.

- virtual void setClosure (void *closure)
- virtual void * getClosure ()
- virtual void destroyTimedWait (long timeout)

  Destroy a queue.

- virtual void destroyWait ()

  Destroy a queue.

- mamaQueue getCValue ()

  Access to C types for implementation of related classes.

- const mamaQueue getCValue () const
- void setCValue (mamaQueue cQueue)

  This can only be set once and only if the c value is not already set - E.g.

Public Attributes

- MamaQueueImpl * mPimpl

7.33.1 Detailed Description

Queue allows applications to dispatch events in order with multiple threads using a single MamaDispatcher for each queue.

7.33.2 Constructor & Destructor Documentation

7.33.2.1 Wombat::MamaQueue::MamaQueue (void)

7.33.2.2 Wombat::MamaQueue::MamaQueue (mamaQueue *cQueue*)

7.33.2.3 virtual Wombat::MamaQueue::~MamaQueue (void) [virtual]

7.33.3 Member Function Documentation

7.33.3.1 virtual void Wombat::MamaQueue::create (mamaBridge *bridgeImpl*) [virtual]

Create a queue.

Queues allow applications to dispatch events in order with multiple threads using a single mamaDispatcher for each queue.

Callers should call delete queue when done.

Returns:

a pointer the queue.

7.33.3.2 virtual void Wombat::MamaQueue::create (mamaBridge *bridgeImpl*, void * *nativeQueue*) [virtual]

7.33.3.3 virtual void Wombat::MamaQueue::dispatch () [virtual]

Dispatch message.

Blocks and dispatches messages until unblock is called.

7.33.3.4 virtual void Wombat::MamaQueue::timedDispatch (uint64_t *timeout*) [virtual]

Dispatch messages until timeout (see release notes for details).

7.33.3.5 virtual void Wombat::MamaQueue::dispatchEvent () [virtual]

Dispatch a single event from the specified queue.

If there is no event on the queue simply return and do nothing

7.33.3.6 virtual void Wombat::MamaQueue::enqueueEvent (MamaQueueEventCallback * *callback*, void * *closure*) [virtual]

Add a user event to a queue.

Parameters:

callback Instance of the MamaQueueEventCallback interface. MamaQueue-EventCallback.onEvent() will be invoked when the event fires.

closure Optional user supplied arbitrary closure data which will be passed back in the MamaQueueEventCallback.onEvent) callback

Exceptions:

MamaException Not currently implemented for pure Java API.

7.33.3.7 virtual void Wombat::MamaQueue::enqueueEvent (MamaQueueEventCallback & *callback*, void * *closure*) [virtual]

7.33.3.8 virtual void Wombat::MamaQueue::stopDispatch () [virtual]

stopDispatch the queue.

7.33.3.9 virtual size_t Wombat::MamaQueue::getEventCount () [virtual]

Returns the number of events currently on the queue.

Returns:

size_t The number of the events on the queue.

7.33.3.10 virtual void Wombat::MamaQueue::setEnqueueCallback (MamaQueueEnqueueCallback * *cb*, void * *closure*) [virtual]

Set a callback which will be invoked as each event is added to the underlying event queue.

Parameters:

cb Pointer to an instance of MamaQueueEnqueueCallback

closure Arbitrary user supplied data. Passed back to onEventEnqueue() for each event enqueued.

7.33.3.11 virtual void Wombat::MamaQueue::setQueueMonitorCallback (MamaQueueMonitorCallback * *cb*, void * *closure*) [virtual]

Register an object to receive callbacks for monitoring the behaviour of the MamaQueue.

Parameters:

cb Reference to the object which will receive callbacks.

closure User supplied data which will be returned as the callbacks are invoked.

7.33.3.12 virtual void Wombat::MamaQueue::setHighWatermark (size_t *highWatermark*) [virtual]

Specify a high watermark for events on the queue.

The behaviour for setting this value varies depending on the underlying middleware.

LBM: LBM uses an unbounded event queue. Setting this values allows users of the API to receive a callback if the value is exceeded. (See mamaQueue_setQueueMonitorCallback() for setting queue related callbacks) the default behaviour is for the queue to grow unbounded without notifications. The high watermark for LBM can be set for all queues at once by setting the mama.lbm.eventqueuemonitor.queue_size_warning property for the API. Calls to this function will override the value specified in mama.properties.

RV: This will set a queue limit policy of TIBRVQUEUE_DISCARD_FIRST whereby the oldest events in the queue are discarded first. The discard amount will be set with a value of 1. i.e. events will be dropped from the queue one at a time. The default behaviour is an unlimited queue which does not discard events.

7.33.3.13 virtual size_t Wombat::MamaQueue::getHighWatermark (void) const [virtual]

Return the high water mark as set via `setHighWaterMark()`.

7.33.3.14 virtual void Wombat::MamaQueue::setLowWatermark (size_t *lowWatermark*) [virtual]

Set the low watermark.

Only supported for Wombat TCP middleware.

Parameters:

lowWatermark The low water mark.

7.33.3.15 virtual size_t Wombat::MamaQueue::getLowWatermark (void) const [virtual]

Return the low water mark as set via `setLowWaterMark()`.

7.33.3.16 virtual void Wombat::MamaQueue::setQueueName (const char * *name*) [virtual]

Associate a name identifier with the event queue.

This will be used in queue related logging statements. The string is copied by the API.

Parameters:

name The string name identifier for the queue.

7.33.3.17 virtual const char* Wombat::MamaQueue::getQueueName () const [virtual]

Retrieve the string name identifier for the queue as specified from a call to `setQueueName()`.

If a name has not been specified via a call to`setQueueName()` the queue will assume a default name of "NO_NAME"

Returns:

The name identifier for the MamaQueue.

7.33.3.18 virtual const char* Wombat::MamaQueue::getQueueBridgeName () const [virtual]

Retrieve the string name identifier for the queue's bridge.

Returns:

The name identifier for the bridge: "wmw", "lbm", or "tibrv".

7.33.3.19 virtual void Wombat::MamaQueue::destroy () [virtual]

Destroy a queue.

Note that the queue can only be destroyed if all of the objects created on it, (timers, subscriptions etc), have been destroyed.

Parameters:

queue The queue.

Exceptions:

MamaStatus with a code of MAMA_STATUS_QUEUE_OPEN_OBJECTS if there are still objects open against the queue.

7.33.3.20 virtual void Wombat::MamaQueue::setClosure (void * *closure*) [virtual]

7.33.3.21 virtual void* Wombat::MamaQueue::getClosure () [virtual]

7.33.3.22 virtual void Wombat::MamaQueue::destroyTimedWait (long *timeout*) [virtual]

Destroy a queue.

Note that the queue can only be destroyed if all of the objects created on it, (timers, subscriptions etc), have been destroyed. This function will block for the specified time or until all of the objects have been destroyed and will then destroy the queue.

Parameters:

timeout The time to block for in ms.

Exceptions:

MamaStatus with a code of MAMA_STATUS_TIMEOUT if the time elapsed.

7.33.3.23 virtual void Wombat::MamaQueue::destroyWait () [virtual]

Destroy a queue.

Note that the queue can only be destroyed if all of the objects created on it, (timers, subscriptions etc), have been destroyed. This function will block until all of the objects have been destroyed and will then destroy the queue.

7.33.3.24 mamaQueue Wombat::MamaQueue::getCValue ()

Access to C types for implementation of related classes.

7.33.3.25 const mamaQueue Wombat::MamaQueue::getCValue () const

7.33.3.26 void Wombat::MamaQueue::setCValue (mamaQueue *cQueue*)

This can only be set once and only if the c value is not already set - E.g.

from calling create()

7.33.4 Member Data Documentation

7.33.4.1 MamaQueueImpl* Wombat::MamaQueue::mPimpl

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

- MamaQueue.h

7.34 Wombat::MamaQueueEnqueueCallback Class Reference

Callback interface for the MamaQueue::setEnqueueCallback () method.

`#include <MamaQueueEnqueueCallback.h>`

Public Member Functions

- virtual ~MamaQueueEnqueueCallback ()
- virtual void onEventEnqueue (void *closure)=0

  Called whenever an event is enqueued to the event queue.

7.34.1 Detailed Description

Callback interface for the MamaQueue::setEnqueueCallback () method.

7.34.2 Constructor & Destructor Documentation

7.34.2.1 virtual Wombat::MamaQueueEnqueueCallback::~MamaQueueEnqueueCallback () [virtual]

```
35         {
36         };
```

7.34.3 Member Function Documentation

7.34.3.1 virtual void Wombat::MamaQueueEnqueueCallback::onEventEnqueue (void * *closure*) [pure virtual]

Called whenever an event is enqueued to the event queue.

LBM Bridge: NB! Users may not dispatch events from this method. The function is invoked from an LBM internal thread. Attempts to dispatch from here will result in a deadlock

Parameters:

closure Arbitrary user-supplied data passed to MamaQueue::setEnqueueCallback ();

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

- MamaQueueEnqueueCallback.h

7.35 Wombat::MamaQueueEventCallback Class Reference

Definition of the callback method for when a user added event fires.

`#include <MamaQueueEventCallback.h>`

Public Member Functions

- virtual ~MamaQueueEventCallback ()
- virtual void onEvent (MamaQueue &queue, void *closure)=0

  Invoked when a user event, added by `MamaQueue.enqueueEvent()` *fires.*

7.35.1 Detailed Description

Definition of the callback method for when a user added event fires.

Concrete instances of this interface are registered with an event queue using the `MamaQueue.enqueueEvent()`. Currently only support by Wombat Middleware.

7.35.2 Constructor & Destructor Documentation

7.35.2.1 virtual Wombat::MamaQueueEventCallback::~MamaQueueEventCallback () [virtual]

```
39 {};
```

7.35.3 Member Function Documentation

7.35.3.1 virtual void Wombat::MamaQueueEventCallback::onEvent (MamaQueue & *queue*, void * *closure*) [pure virtual]

Invoked when a user event, added by `MamaQueue.enqueueEvent()` fires.

Parameters:

queue The MamaQueue on which the event was enqueued.

closure The user specified data associated with this event.

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

- MamaQueueEventCallback.h

7.36 Wombat::MamaQueueGroup Class Reference

A simple class for allocating subscriptions amongst multiple queues in a round robin.

`#include <MamaQueueGroup.h>`

Public Member Functions

- virtual ~MamaQueueGroup ()
- MamaQueueGroup (int numberOfQueues, mamaBridge bridgeImpl)

  If numberOfQueues == 0, getNextQueue returns the default queue for the bridge.

- virtual void destroyWait ()

  Destroy all the queues.

- virtual MamaQueue * getNextQueue ()

  Return the next available queue from the queue group.

- virtual int getNumberOfQueues ()

  Return the number of MamaQueues currently managed by this queue group.

- virtual void stopDispatch ()

  Stop dispatching on queues in the queue group.

- virtual void startDispatch ()

  Start dispatching on all queues in a group.

7.36.1 Detailed Description

A simple class for allocating subscriptions amongst multiple queues in a round robin.

This class creates dispatchers for the queues as well.

7.36.2 Constructor & Destructor Documentation

7.36.2.1 virtual Wombat::MamaQueueGroup::~MamaQueueGroup () [virtual]

7.36.2.2 Wombat::MamaQueueGroup::MamaQueueGroup (int *numberOfQueues*, mamaBridge *bridgeImpl*)

If numberOfQueues == 0, getNextQueue returns the default queue for the bridge.

7.36.3 Member Function Documentation

7.36.3.1 virtual void Wombat::MamaQueueGroup::destroyWait ()

[virtual]

Destroy all the queues.

Note that a queue can only be destroyed if all of the objects created on it, (timers, subscriptions etc), have been destroyed. This function will block until all of the objects have been destroyed and will then destroy the queues.

7.36.3.2 virtual MamaQueue* Wombat::MamaQueueGroup::getNextQueue ()

[virtual]

Return the next available queue from the queue group.

Queues are returned in a round robin fashion.

7.36.3.3 virtual int Wombat::MamaQueueGroup::getNumberOfQueues ()

[virtual]

Return the number of MamaQueues currently managed by this queue group.

7.36.3.4 virtual void Wombat::MamaQueueGroup::stopDispatch ()

[virtual]

Stop dispatching on queues in the queue group.

7.36.3.5 virtual void Wombat::MamaQueueGroup::startDispatch ()

[virtual]

Start dispatching on all queues in a group.

NOTE: This only should be used after a previous call to stopDispatch. Dispatching on a queue is started hwen it is created

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

- MamaQueueGroup.h

7.37 Wombat::MamaQueueMonitorCallback Class Reference

Receive callbacks when certain conditions for the MamaQueue are met.

`#include <MamaQueueMonitorCallback.h>`

Public Member Functions

- virtual ~MamaQueueMonitorCallback ()
- virtual void onHighWatermarkExceeded (MamaQueue *queue, size_t size, void *closure)=0

  Callback invoked if an upper size limit has been specified for a queue and that limit has been exceeded.

- virtual void onLowWatermark (MamaQueue *queue, size_t size, void *closure)=0

  Callback when low water mark is reached.

7.37.1 Detailed Description

Receive callbacks when certain conditions for the MamaQueue are met.

Currently only one callback is defined which is invoked when the specified size limit on the MamaQueue is exceeded.

7.37.2 Constructor & Destructor Documentation

7.37.2.1 virtual Wombat::MamaQueueMonitorCallback::~MamaQueueMonitorCallback () [virtual]

```
38              {
39              }
```

7.37.3 Member Function Documentation

7.37.3.1 virtual void Wombat::MamaQueueMonitorCallback::onHighWatermarkExceeded (MamaQueue * *queue*, size_t *size*, void * *closure*) [pure virtual]

Callback invoked if an upper size limit has been specified for a queue and that limit has been exceeded.

Parameters:

queue Pointer to the queue for which this callback was invoked.

size The number of events on the queue if supported; otherwise 0.

closure User supplied data when the callback object was registered.

7.37.3.2 virtual void Wombat::MamaQueueMonitorCallback::onLowWatermark (MamaQueue * *queue*, size_t *size*, void * *closure*) [pure virtual]

Callback when low water mark is reached.

Only supported by Wombat TCP middleware.

Parameters:

queue Pointer to the queue for which this callback was invoked.

size The number of events on the queue.

closure User supplied data when the callback object was registered.

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

- MamaQueueMonitorCallback.h

7.38 Wombat::MamaReservedFields Class Reference

`#include <MamaReservedFields.h>`

Static Public Member Functions

- static void initReservedFields ()
- static void uninitReservedFields ()

Static Public Attributes

- static const MamaFieldDescriptor * MsgType
- static const MamaFieldDescriptor * MsgStatus
- static const MamaFieldDescriptor * FieldIndex
- static const MamaFieldDescriptor * MsgNum
- static const MamaFieldDescriptor * MsgTotal
- static const MamaFieldDescriptor * SeqNum
- static const MamaFieldDescriptor * FeedName
- static const MamaFieldDescriptor * FeedHost
- static const MamaFieldDescriptor * FeedGroup
- static const MamaFieldDescriptor * ItemSeqNum
- static const MamaFieldDescriptor * SendTime
- static const MamaFieldDescriptor * AppDataType
- static const MamaFieldDescriptor * AppMsgType
- static const MamaFieldDescriptor * SenderId
- static const MamaFieldDescriptor * MsgQual
- static const MamaFieldDescriptor * ConflateQuoteCount
- static const MamaFieldDescriptor * EntitleCode
- static const MamaFieldDescriptor * SymbolList

7.38.1 Member Function Documentation

7.38.1.1 static void Wombat::MamaReservedFields::initReservedFields () [static]

7.38.1.2 static void Wombat::MamaReservedFields::uninitReservedFields () [static]

7.38.2 Member Data Documentation

7.38.2.1 const MamaFieldDescriptor* Wombat::MamaReservedFields::MsgType [static]

7.38.2.2 const MamaFieldDescriptor* Wombat::MamaReservedFields::MsgStatus [static]

7.38.2.3 const MamaFieldDescriptor* Wombat::MamaReservedFields::FieldIndex [static]

7.38.2.4 const MamaFieldDescriptor* Wombat::MamaReservedFields::MsgNum [static]

7.38.2.5 const MamaFieldDescriptor* Wombat::MamaReservedFields::MsgTotal [static]

7.38.2.6 const MamaFieldDescriptor* Wombat::MamaReservedFields::SeqNum [static]

7.38.2.7 const MamaFieldDescriptor* Wombat::MamaReservedFields::FeedName [static]

7.38.2.8 const MamaFieldDescriptor* Wombat::MamaReservedFields::FeedHost [static]

7.38.2.9 const MamaFieldDescriptor* Wombat::MamaReservedFields::FeedGroup [static]

7.38.2.10 const MamaFieldDescriptor* Wombat::MamaReservedFields::ItemSeqNum [static]

7.38.2.11 const MamaFieldDescriptor* Wombat::MamaReservedFields::SendTime [static]

7.38.2.12 const MamaFieldDescriptor* Wombat::MamaReservedFields::AppDataType [static]

7.38.2.13 const MamaFieldDescriptor* Wombat::MamaReservedFields::AppMsgType [static]

7.38.2.14 const MamaFieldDescriptor* Wombat::MamaReservedFields::SenderId [static]

7.38.2.15 const MamaFieldDescriptor* Wombat::MamaReservedFields::MsgQual [static]

- MamaReservedFields.h

7.39 Wombat::MamaSendCompleteCallback Class Reference

Callback interface for use with the MamaPublisher.sendWithThrottle() and MamaPublisher.sendFromInboxWithThrottle() methods.

`#include <MamaSendCompleteCallback.h>`

Public Member Functions

- virtual ~MamaSendCompleteCallback ()
- virtual void onSendComplete (MamaPublisher &publisher, MamaMsg *msg, MamaStatus &status, void *closure)=0

  Called whenever the API has sent a message on the throttle queue.

7.39.1 Detailed Description

Callback interface for use with the MamaPublisher.sendWithThrottle() and MamaPublisher.sendFromInboxWithThrottle() methods.

The `onSendComplete()` method is invoked once a message being sent on the throttle is no longer required by the API.

Messages sent on the throttle queue are no longer destroyed by the API. It is the responsibility of the application developer to manage the lifecycle of any messages sent on the throttle.

7.39.2 Constructor & Destructor Documentation

7.39.2.1 virtual Wombat::MamaSendCompleteCallback::~MamaSendCompleteCallback () [virtual]

```
46          {
47          };
```

7.39.3 Member Function Documentation

7.39.3.1 virtual void Wombat::MamaSendCompleteCallback::onSendComplete (MamaPublisher & *publisher*, MamaMsg * *msg*, MamaStatus & *status*, void * *closure*) [pure virtual]

Called whenever the API has sent a message on the throttle queue.

Parameters:

publisher The publisher object used to send the message.

msg The MamaMsg which has been sent from the throttle queue.

status Whether the message was successfully sent from the throttle. A value of MAMA_STATUS_OK indicates that the send was successful.

closure User supplied context data.

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

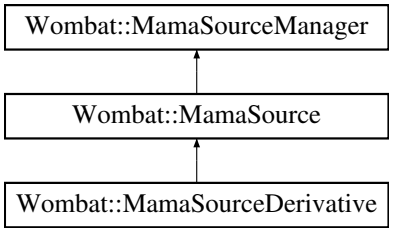
- MamaSendCompleteCallback.h

7.40 Wombat::MamaSource Class Reference

A MAMA source maintains information about a data source, including the quality of the data coming from that source.

`#include <MamaSource.h>`

Inheritance diagram for Wombat::MamaSource::



Public Member Functions

- MamaSource ()
- MamaSource (const char *id, const char *transportName, const char *subscSourceName, mamaBridge bridge, bool createTransport=true)
- MamaSource (const char *id, MamaTransport *transport, const char *subscSourceName)
- virtual ~MamaSource ()
- void setId (const char *id)
- void setMappedId (const char *id)
- void setDisplayId (const char *id)
- void setQuality (mamaQuality quality)
- void setState (mamaSourceState state)
- void setParent (MamaSource *parent)
- void setTransport (MamaTransport *transport)
- void setTransportName (const char *transportName)
- void setPublisherSourceName (const char *sourceName)
- virtual const char * getId () const
- virtual const char * getMappedId () const
- virtual const char * getDisplayId () const
- virtual mamaQuality getQuality () const
- virtual mamaSourceState getState () const
- virtual MamaSource * getParent ()
- virtual const MamaSource * getParent () const
- virtual MamaTransport * getTransport () const
- virtual const char * getTransportName () const

- virtual const char * getPublisherSourceName () const
- mamaSource getCValue ()
- const mamaSource getCValue () const
- bool isPartOf (const MamaSource *source) const

  Is this source part of the supplied source i.e.

- void addSubscription (const char *symbol, MamaSubscription *sub)

  Add a subscription.

- MamaSubscription * findSubscription (const char *symbol)

  Look up a MamaSubscription.

- MamaSubscription * removeSubscription (const char *symbol)

  Remove a MamaSubscription.

- void deactivateSubscriptions ()

  Activate all subscriptions for this source.

- void activateSubscriptions ()

  deactivate all subscriptions for this source.

- void forEachSubscription (MamaSubscriptionIteratorCallback *callback, void *closure)

  Iterate through all subscriptions and call the MamaSubscriptionIterator's onSubscription method for each.

7.40.1 Detailed Description

A MAMA source maintains information about a data source, including the quality of the data coming from that source.

It inherits MamaSourceManager because a source can have sub-sources.

7.40.2 Constructor & Destructor Documentation

7.40.2.1 Wombat::MamaSource::MamaSource ()

7.40.2.2 Wombat::MamaSource::MamaSource (const char * *id*, const char * *transportName*, const char * *subscSourceName*, mamaBridge *bridge*, bool *createTransport* = `true`)

7.40.2.3 Wombat::MamaSource::MamaSource (const char * *id*, MamaTransport * *transport*, const char * *subscSourceName*)

7.40.2.4 virtual Wombat::MamaSource::~MamaSource () `[virtual]`

7.40.3 Member Function Documentation

7.40.3.1 void Wombat::MamaSource::setId (const char * *id*)

7.40.3.2 void Wombat::MamaSource::setMappedId (const char * *id*)

7.40.3.3 void Wombat::MamaSource::setDisplayId (const char * *id*)

7.40.3.4 void Wombat::MamaSource::setQuality (mamaQuality *quality*)

Reimplemented in Wombat::MamaSourceDerivative.

7.40.3.5 void Wombat::MamaSource::setState (mamaSourceState *state*)

Reimplemented in Wombat::MamaSourceDerivative.

7.40.3.6 void Wombat::MamaSource::setParent (MamaSource * *parent*)

7.40.3.7 void Wombat::MamaSource::setTransport (MamaTransport * *transport*)

7.40.3.8 void Wombat::MamaSource::setTransportName (const char * *transportName*)

7.40.3.9 void Wombat::MamaSource::setPublisherSourceName (const char * *sourceName*)

7.40.3.10 virtual const char* Wombat::MamaSource::getId () const [virtual]

7.40.3.11 virtual const char* Wombat::MamaSource::getMappedId () const [virtual]

7.40.3.12 virtual const char* Wombat::MamaSource::getDisplayId () const [virtual]

7.40.3.13 virtual mamaQuality Wombat::MamaSource::getQuality () const [virtual]

Reimplemented in Wombat::MamaSourceDerivative.

7.40.3.14 virtual mamaSourceState Wombat::MamaSource::getState () const [virtual]

Reimplemented in Wombat::MamaSourceDerivative.

7.40.3.15 virtual MamaSource* Wombat::MamaSource::getParent () [virtual]

7.40.3.16 virtual const MamaSource* Wombat::MamaSource::getParent () const [virtual]

7.40.3.17 virtual MamaTransport* Wombat::MamaSource::getTransport () const [virtual]

7.40.3.18 virtual const char* Wombat::MamaSource::getTransportName () const [virtual]

7.40.3.19 virtual const char* Wombat::MamaSource::getPublisherSourceName () const [virtual]

7.40.3.20 mamaSource Wombat::MamaSource::getCValue ()

Reimplemented from Wombat::MamaSourceManager.

7.40.3.21 const mamaSource Wombat::MamaSource::getCValue () const

Reimplemented from Wombat::MamaSourceManager.

7.40.3.22 bool Wombat::MamaSource::isPartOf (const MamaSource * *source*) const

Is this source part of the supplied source i.e.

Is it the same as the supplied source or is the supplied source a parent (or parent of a parent) of this source

7.40.3.23 void Wombat::MamaSource::addSubscription (const char * *symbol*, MamaSubscription * *sub*)

Add a subscription.

7.40.3.24 MamaSubscription* Wombat::MamaSource::findSubscription (const char * *symbol*)

Look up a MamaSubscription.

NULL is returned if not found.

7.40.3.25 MamaSubscription* Wombat::MamaSource::removeSubscription (const char * *symbol*)

Remove a MamaSubscription.

Subscription return is the subscription removed from list of associated subscriptions. NULL is returned if not found.

7.40.3.26 void Wombat::MamaSource::deactivateSubscriptions ()

Activate all subscriptions for this source.

7.40.3.27 void Wombat::MamaSource::activateSubscriptions ()

deactivate all subscriptions for this source.

7.40.3.28 void Wombat::MamaSource::forEachSubscription (MamaSubscriptionIteratorCallback * *callback*, void * *closure*)

Iterate through all subscriptions and call the MamaSubscriptionIterator's onSubscription method for each.

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

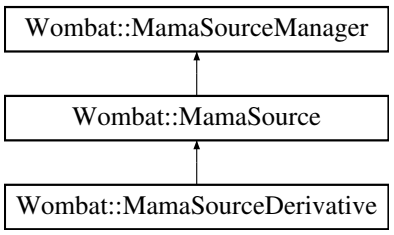
- MamaSource.h

7.41 Wombat::MamaSourceDerivative Class Reference

A MamaSourceDerivative provides a reference to a common MamaSource object but can have attributes (such as the quality state) overridden.

`#include <MamaSourceDerivative.h>`

Inheritance diagram for Wombat::MamaSourceDerivative::



Public Member Functions

- MamaSourceDerivative (const MamaSource *baseSource)

  Construct an instance of a derived MAMA source.

- virtual ~MamaSourceDerivative ()
- virtual void setQuality (mamaQuality quality)
- virtual void setState (mamaSourceState state)
- virtual mamaQuality getQuality () const
- virtual mamaSourceState getState () const
- virtual MamaSourceDerivative * find (const char *sourceName)
- virtual const MamaSourceDerivative * find (const char *sourceName) const
- const MamaSource * getBaseSource () const

7.41.1 Detailed Description

A MamaSourceDerivative provides a reference to a common MamaSource object but can have attributes (such as the quality state) overridden.

This class is intended to be associated with individually subscribed items, including order books.

7.41.2 Constructor & Destructor Documentation

7.41.2.1 Wombat::MamaSourceDerivative::MamaSourceDerivative (const MamaSource * *baseSource*)

Construct an instance of a derived MAMA source.

The resulting source derivative will have derived sub-sources for each of the sub-sources in baseSource.

7.41.2.2 virtual Wombat::MamaSourceDerivative::~MamaSourceDerivative () [virtual]

7.41.3 Member Function Documentation

7.41.3.1 virtual void Wombat::MamaSourceDerivative::setQuality (mamaQuality *quality*) [virtual]

Reimplemented from Wombat::MamaSource.

7.41.3.2 virtual void Wombat::MamaSourceDerivative::setState (mamaSourceState *state*) [virtual]

Reimplemented from Wombat::MamaSource.

7.41.3.3 virtual mamaQuality Wombat::MamaSourceDerivative::getQuality () const [virtual]

Reimplemented from Wombat::MamaSource.

7.41.3.4 virtual mamaSourceState Wombat::MamaSourceDerivative::getState () const [virtual]

Reimplemented from Wombat::MamaSource.

7.41.3.5 virtual MamaSourceDerivative* Wombat::MamaSourceDerivative::find (const char * *sourceName*) [virtual]

Reimplemented from Wombat::MamaSourceManager.

7.41.3.6 virtual const MamaSourceDerivative* Wombat::MamaSourceDerivative::find (const char * *sourceName*) const [virtual]

Reimplemented from Wombat::MamaSourceManager.

7.41.3.7 const MamaSource* Wombat::MamaSourceDerivative::getBaseSource () const

```
59          {
60              return myBaseSource;
61          }
```

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

- MamaSourceDerivative.h

7.42 Wombat::MamaSourceGroup Class Reference

A MAMA source group monitors a set of MAMA sources that presumably provide a duplicate set of data.

`#include <MamaSourceGroup.h>`

Public Types

- typedef const iterator const_iterator

Public Member Functions

- MamaSourceGroup (const char *name)
- ~MamaSourceGroup ()
- const char * getName () const
- MamaSource * find (const char *sourceName)
- const MamaSource * find (const char *sourceName) const
- void add (MamaSource *source, mama_u32_t weight)
- void add (const char *sourceName, mama_u32_t weight, MamaSource *source)
- void setWeight (const char *sourceName, mama_u32_t weight)
- mama_u32_t getWeight (const char *sourceName) const
- mama_size_t size () const
- void registerStateChangeCallback (MamaSourceStateChangeCallback &cb)

  Applications interested in event notifications can register for events.

- void unregisterStateChangeCallback (MamaSourceStateChangeCallback &cb)

  Applications interested in event notifications can unregister for events.

- bool reevaluate ()

  Re-evaluate the group by checking all of the relative weights and changing the state of each MamaSource in the group as appropriate.

- iterator begin ()
- const_iterator begin () const
- iterator end ()
- const_iterator end () const

Classes

- class iterator

7.42.1 Detailed Description

A MAMA source group monitors a set of MAMA sources that presumably provide a duplicate set of data.

Each member of the group is given a priority and the member with the highest priority is given the mamaSourceState, MAMA_SOURCE_STATE_OK; the other members of the group are given the state, MAMA_SOURCE_STATE_OFF.

7.42.2 Member Typedef Documentation

7.42.2.1 typedef const iterator Wombat::MamaSourceGroup::const_iterator

7.42.3 Constructor & Destructor Documentation

7.42.3.1 Wombat::MamaSourceGroup::MamaSourceGroup (const char * *name*)

7.42.3.2 Wombat::MamaSourceGroup::~MamaSourceGroup ()

7.42.4 Member Function Documentation

7.42.4.1 const char* Wombat::MamaSourceGroup::getName () const

7.42.4.2 MamaSource* Wombat::MamaSourceGroup::find (const char * *sourceName*)

7.42.4.3 const MamaSource* Wombat::MamaSourceGroup::find (const char * *sourceName*) const

7.42.4.4 void Wombat::MamaSourceGroup::add (MamaSource * *source*, mama_u32_t *weight*)

7.42.4.5 void Wombat::MamaSourceGroup::add (const char * *sourceName*, mama_u32_t *weight*, MamaSource * *source*)

7.42.4.6 void Wombat::MamaSourceGroup::setWeight (const char * *sourceName*, mama_u32_t *weight*)

7.42.4.7 mama_u32_t Wombat::MamaSourceGroup::getWeight (const char * *sourceName*) const

7.42.4.8 mama_size_t Wombat::MamaSourceGroup::size () const

7.42.4.9 void Wombat::MamaSourceGroup::registerStateChangeCallback (MamaSourceStateChangeCallback & *cb*)

Applications interested in event notifications can register for events.

Parameters:

cb callback to register

7.42.4.10 void Wombat::MamaSourceGroup::unregisterStateChangeCallback (MamaSourceStateChangeCallback & *cb*)

Applications interested in event notifications can unregister for events.

Parameters:

cb callback to unregister

7.42.4.11 bool Wombat::MamaSourceGroup::reevaluate ()

Re-evaluate the group by checking all of the relative weights and changing the state of each MamaSource in the group as appropriate.

Returns true if any states were changed; otherwise false.

7.42.4.12 iterator Wombat::MamaSourceGroup::begin ()

7.42.4.13 const_iterator Wombat::MamaSourceGroup::begin () const

7.42.4.14 iterator Wombat::MamaSourceGroup::end ()

7.42.4.15 const_iterator Wombat::MamaSourceGroup::end () const

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

- MamaSourceGroup.h

7.43 Wombat::MamaSourceGroup::iterator Class Reference

`#include <MamaSourceGroup.h>`

Public Member Functions

- iterator ()
- iterator (const iterator &copy)
- iterator (const iteratorImpl &copy)
- ~iterator ()
- iterator & operator= (const iterator &rhs)
- const iterator & operator= (const iterator &rhs) const
- iterator & operator++ ()
- const iterator & operator++ () const
- bool operator== (const iterator &rhs) const
- bool operator!= (const iterator &rhs) const
- MamaSource * operator * ()
- const MamaSource * operator * () const

Protected Attributes

- iteratorImpl & mImpl

Friends

- class MamaSourceGroup

7.43.1 Constructor & Destructor Documentation

7.43.1.1 Wombat::MamaSourceGroup::iterator::iterator ()

7.43.1.2 Wombat::MamaSourceGroup::iterator::iterator (const iterator & *copy*)

7.43.1.3 Wombat::MamaSourceGroup::iterator::iterator (const iteratorImpl & *copy*)

7.43.1.4 Wombat::MamaSourceGroup::iterator::~iterator ()

7.43.2 Member Function Documentation

7.43.2.1 iterator& Wombat::MamaSourceGroup::iterator::operator= (const iterator & *rhs*)

7.43.2.2 const iterator& Wombat::MamaSourceGroup::iterator::operator= (const iterator & *rhs*) const

7.43.2.3 iterator& Wombat::MamaSourceGroup::iterator::operator++ ()

7.43.2.4 const iterator& Wombat::MamaSourceGroup::iterator::operator++ () const

7.43.2.5 bool Wombat::MamaSourceGroup::iterator::operator== (const iterator & *rhs*) const

7.43.2.6 bool Wombat::MamaSourceGroup::iterator::operator!= (const iterator & *rhs*) const

7.43.2.7 MamaSource* Wombat::MamaSourceGroup::iterator::operator * ()

7.43.2.8 const MamaSource* Wombat::MamaSourceGroup::iterator::operator * () const

7.43.3 Friends And Related Function Documentation

7.43.3.1 friend class MamaSourceGroup [friend]

7.43.4 Member Data Documentation

7.43.4.1 iteratorImpl& Wombat::MamaSourceGroup::iterator::mImpl [protected]

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

- MamaSourceGroup.h

7.44 Wombat::MamaSourceGroupManager Class Reference

A MAMA source group manager monitors a set of MAMA source groups.

`#include <MamaSourceGroupManager.h>`

Public Types

- typedef const iterator const_iterator

Public Member Functions

- MamaSourceGroupManager ()
- ~MamaSourceGroupManager ()
- MamaSourceGroup * create (const char *groupName)
- MamaSourceGroup * findOrCreate (const char *groupName)
- MamaSourceGroup * find (const char *groupName)
- const MamaSourceGroup * find (const char *groupName) const
- mama_size_t size () const
- iterator begin ()
- const_iterator begin () const
- iterator end ()
- const_iterator end () const

Classes

- class iterator

7.44.1 Detailed Description

A MAMA source group manager monitors a set of MAMA source groups.

7.44.2 Member Typedef Documentation

7.44.2.1 typedef const iterator Wombat::MamaSourceGroupManager::const_iterator

7.44.3 Constructor & Destructor Documentation

7.44.3.1 Wombat::MamaSourceGroupManager::MamaSourceGroupManager ()

7.44.3.2 Wombat::MamaSourceGroupManager::~MamaSourceGroupManager ()

7.44.4 Member Function Documentation

7.44.4.1 MamaSourceGroup* Wombat::MamaSourceGroupManager::create (const char * *groupName*)

7.44.4.2 MamaSourceGroup* Wombat::MamaSourceGroupManager::findOrCreate (const char * *groupName*)

7.44.4.3 MamaSourceGroup* Wombat::MamaSourceGroupManager::find (const char * *groupName*)

7.44.4.4 const MamaSourceGroup* Wombat::MamaSourceGroupManager::find (const char * *groupName*) const

7.44.4.5 mama_size_t Wombat::MamaSourceGroupManager::size () const

7.44.4.6 iterator Wombat::MamaSourceGroupManager::begin ()

7.44.4.7 const_iterator Wombat::MamaSourceGroupManager::begin () const

7.44.4.8 iterator Wombat::MamaSourceGroupManager::end ()

7.44.4.9 const_iterator Wombat::MamaSourceGroupManager::end () const

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

- MamaSourceGroupManager.h

7.45 Wombat::MamaSourceGroupManager::iterator Class Reference

`#include <MamaSourceGroupManager.h>`

Public Member Functions

- iterator ()
- iterator (const iterator &copy)
- iterator (const iteratorImpl &copy)
- ~iterator ()
- iterator & operator= (const iterator &rhs)
- const iterator & operator= (const iterator &rhs) const
- iterator & operator++ ()
- const iterator & operator++ () const
- bool operator== (const iterator &rhs) const
- bool operator!= (const iterator &rhs) const
- MamaSourceGroup * operator * ()
- const MamaSourceGroup * operator * () const

Protected Attributes

- iteratorImpl & mImpl

Friends

- class MamaSourceGroupManager

7.45.1 Constructor & Destructor Documentation

7.45.1.1 Wombat::MamaSourceGroupManager::iterator::iterator ()

7.45.1.2 Wombat::MamaSourceGroupManager::iterator::iterator (const iterator & *copy*)

7.45.1.3 Wombat::MamaSourceGroupManager::iterator::iterator (const iteratorImpl & *copy*)

7.45.1.4 Wombat::MamaSourceGroupManager::iterator::~iterator ()

7.45.2 Member Function Documentation

7.45.2.1 iterator& Wombat::MamaSourceGroupManager::iterator::operator= (const iterator & *rhs*)

7.45.2.2 const iterator& Wombat::MamaSourceGroupManager::iterator::operator= (const iterator & *rhs*) const

7.45.2.3 iterator& Wombat::MamaSourceGroupManager::iterator::operator++ ()

7.45.2.4 const iterator& Wombat::MamaSourceGroupManager::iterator::operator++ () const

7.45.2.5 bool Wombat::MamaSourceGroupManager::iterator::operator== (const iterator & *rhs*) const

7.45.2.6 bool Wombat::MamaSourceGroupManager::iterator::operator!= (const iterator & *rhs*) const

7.45.2.7 MamaSourceGroup* Wombat::MamaSourceGroupManager::iterator::operator * ()

7.45.2.8 const MamaSourceGroup* Wombat::MamaSourceGroupManager::iterator::operator * () const

7.45.3 Friends And Related Function Documentation

7.45.3.1 friend class MamaSourceGroupManager [friend]

7.45.4 Member Data Documentation

7.45.4.1 iteratorImpl& Wombat::MamaSourceGroupManager::iterator::m_impl

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

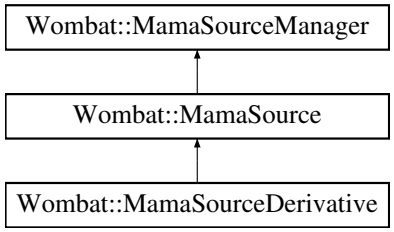
- MamaSourceGroupManager.h

7.46 Wombat::MamaSourceManager Class Reference

A MAMA source manager maintains information about a set of data sources, including the quality of the data coming from those sources.

`#include <MamaSourceManager.h>`

Inheritance diagram for Wombat::MamaSourceManager::



Public Types

- typedef const iterator const_iterator

Public Member Functions

- MamaSourceManager ()
- virtual ~MamaSourceManager ()
- virtual MamaSource * create (const char *sourceName)
- virtual MamaSource * findOrCreate (const char *sourceName)
- virtual MamaSource * find (const char *sourceName)
- virtual const MamaSource * find (const char *sourceName) const
- virtual void add (MamaSource *source)
- virtual void add (const char *sourceName, MamaSource *source)
- mama_size_t size () const
- iterator begin ()
- const_iterator begin () const
- iterator end ()
- const_iterator end () const
- mamaSourceManager getCValue ()
- const mamaSourceManager getCValue () const

Classes

- class iterator

7.46.1 Detailed Description

A MAMA source manager maintains information about a set of data sources, including the quality of the data coming from those sources.

7.46.2 Member Typedef Documentation

7.46.2.1 typedef const iterator Wombat::MamaSourceManager::const_iterator

7.46.3 Constructor & Destructor Documentation

7.46.3.1 Wombat::MamaSourceManager::MamaSourceManager ()

7.46.3.2 virtual Wombat::MamaSourceManager::~MamaSourceManager () [virtual]

7.46.4 Member Function Documentation

7.46.4.1 virtual MamaSource* Wombat::MamaSourceManager::create (const char * *sourceName*) [virtual]

7.46.4.2 virtual MamaSource* Wombat::MamaSourceManager::findOrCreate (const char * *sourceName*) [virtual]

7.46.4.3 virtual MamaSource* Wombat::MamaSourceManager::find (const char * *sourceName*) [virtual]

Reimplemented in Wombat::MamaSourceDerivative.

7.46.4.4 virtual const MamaSource* Wombat::MamaSourceManager::find (const char * *sourceName*) const [virtual]

Reimplemented in Wombat::MamaSourceDerivative.

7.46.4.5 virtual void Wombat::MamaSourceManager::add (MamaSource * *source*) [virtual]

7.46.4.6 virtual void Wombat::MamaSourceManager::add (const char * *sourceName*, MamaSource * *source*) [virtual]

7.46.4.7 mama_size_t Wombat::MamaSourceManager::size () const

7.46.4.8 iterator Wombat::MamaSourceManager::begin ()

7.46.4.9 const_iterator Wombat::MamaSourceManager::begin () const

7.46.4.10 iterator Wombat::MamaSourceManager::end ()

7.46.4.11 const_iterator Wombat::MamaSourceManager::end () const

7.46.4.12 mamaSourceManager Wombat::MamaSourceManager::getCValue ()

Reimplemented in Wombat::MamaSource.

```
85          {
86              return myManager;
87          }
```

7.46.4.13 const mamaSourceManager Wombat::MamaSourceManager::getCValue () const

Reimplemented in Wombat::MamaSource.

```
90          {
91              return myManager;
92          }
```

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

- MamaSourceManager.h

7.47 Wombat::MamaSourceManager::iterator Class Reference

`#include <MamaSourceManager.h>`

Public Member Functions

- iterator ()
- iterator (const iterator &copy)
- iterator (const iteratorImpl &copy)
- ~iterator ()
- iterator & operator= (const iterator &rhs)
- const iterator & operator= (const iterator &rhs) const
- iterator & operator++ ()
- const iterator & operator++ () const
- bool operator== (const iterator &rhs) const
- bool operator!= (const iterator &rhs) const
- MamaSource * operator * ()
- const MamaSource * operator * () const

Protected Attributes

- iteratorImpl & mImpl

Friends

- class MamaSourceManager

7.47.1 Constructor & Destructor Documentation

7.47.1.1 Wombat::MamaSourceManager::iterator::iterator ()

7.47.1.2 Wombat::MamaSourceManager::iterator::iterator (const iterator & *copy*)

7.47.1.3 Wombat::MamaSourceManager::iterator::iterator (const iteratorImpl & *copy*)

7.47.1.4 Wombat::MamaSourceManager::iterator::~iterator ()

7.47.2 Member Function Documentation

7.47.2.1 iterator& Wombat::MamaSourceManager::iterator::operator= (const iterator & *rhs*)

7.47.2.2 const iterator& Wombat::MamaSourceManager::iterator::operator= (const iterator & *rhs*) const

7.47.2.3 iterator& Wombat::MamaSourceManager::iterator::operator++ ()

7.47.2.4 const iterator& Wombat::MamaSourceManager::iterator::operator++ () const

7.47.2.5 bool Wombat::MamaSourceManager::iterator::operator== (const iterator & *rhs*) const

7.47.2.6 bool Wombat::MamaSourceManager::iterator::operator!= (const iterator & *rhs*) const

7.47.2.7 MamaSource* Wombat::MamaSourceManager::iterator::operator * ()

7.47.2.8 const MamaSource* Wombat::MamaSourceManager::iterator::operator * () const

7.47.3 Friends And Related Function Documentation

7.47.3.1 friend class MamaSourceManager [friend]

7.47.4 Member Data Documentation

7.47.4.1 iteratorImpl& Wombat::MamaSourceManager::iterator::mImpl [protected]

- MamaSourceManager.h

7.48 Wombat::MamaSourceStateChangeCallback Class Reference

Applications can register with MamaSourceGroup to receive state change notifications when the state of sources within the group has changed.

`#include <MamaSourceStateChangeCallback.h>`

Public Member Functions

- virtual ~MamaSourceStateChangeCallback (void)
- virtual void onStateChanged (MamaSourceGroup &sourceGroup, MamaSource *topWeightSource)=0

  State change callback.

7.48.1 Detailed Description

Applications can register with MamaSourceGroup to receive state change notifications when the state of sources within the group has changed.

7.48.2 Constructor & Destructor Documentation

7.48.2.1 virtual Wombat::MamaSourceStateChangeCallback::~MamaSourceStateChangeCallback (void) [virtual]

```
43         {}
```

7.48.3 Member Function Documentation

7.48.3.1 virtual void Wombat::MamaSourceStateChangeCallback::onStateChanged (MamaSourceGroup & *sourceGroup*, MamaSource * *topWeightSource*) [pure virtual]

State change callback.

Parameters:

sourceGroup The source group notifying the applications.

topWeightSource The top weight source in the group. Determined when _reevaluate() is called.

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

- MamaSourceStateChangeCallback.h

7.49 Wombat::MamaStartCallback Class Reference

Callback object passed to Mama::startBackground().

`#include <mamacpp.h>`

Public Member Functions

- virtual ~MamaStartCallback ()
- virtual void onStartComplete (MamaStatus status)=0

7.49.1 Detailed Description

Callback object passed to Mama::startBackground().

The onStartComplete() method will be invoked if an error occurs calling Mama::startBackground() or when Mama::startBackground() exits normally in which case the status returned will be MAMA_STATUS_OK

7.49.2 Constructor & Destructor Documentation

7.49.2.1 virtual Wombat::MamaStartCallback::~MamaStartCallback () [virtual]

```
133 {};
```

7.49.3 Member Function Documentation

7.49.3.1 virtual void Wombat::MamaStartCallback::onStartComplete (MamaStatus *status*) [pure virtual]

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

- mamacpp.h

7.50 Wombat::MamaStat Class Reference

`#include <MamaStat.h>`

Public Member Functions

- MamaStat (void)
- virtual ~MamaStat (void)
- virtual void create (MamaStatsCollector *statsCollector, int lockable, const char *name, mama_fid_t fid)
- virtual void increment ()
- virtual void decrement ()
- virtual void reset ()
- virtual void setLog (int log)
- virtual void setPublish (int publish)
- virtual void destroy (void)

Protected Member Functions

- MamaStat (MamaStatImpl *)

Protected Attributes

- MamaStatImpl * mSimpl

7.50.1 Constructor & Destructor Documentation

7.50.1.1 Wombat::MamaStat::MamaStat (void)

7.50.1.2 virtual Wombat::MamaStat::~MamaStat (void) [virtual]

7.50.1.3 Wombat::MamaStat::MamaStat (MamaStatImpl *) [protected]

7.50.2 Member Function Documentation

7.50.2.1 virtual void Wombat::MamaStat::create (MamaStatsCollector * *statsCollector*, int *lockable*, const char * *name*, mama_fid_t *fid*) [virtual]

7.50.2.2 virtual void Wombat::MamaStat::increment () [virtual]

7.50.2.3 virtual void Wombat::MamaStat::decrement () [virtual]

7.50.2.4 virtual void Wombat::MamaStat::reset () [virtual]

7.50.2.5 virtual void Wombat::MamaStat::setLog (int *log*) [virtual]

7.50.2.6 virtual void Wombat::MamaStat::setPublish (int *publish*) [virtual]

7.50.2.7 virtual void Wombat::MamaStat::destroy (void) [virtual]

7.50.3 Member Data Documentation

7.50.3.1 MamaStatImpl* Wombat::MamaStat::mSimpl [protected]

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

- MamaStat.h

7.51 Wombat::MamaStatsCollector Class Reference

`#include <MamaStatsCollector.h>`

Public Member Functions

- MamaStatsCollector (void)
- virtual ~MamaStatsCollector (void)
- virtual void create (mamaStatsCollectorType type, const char *name, const char *middleware)
- virtual void incrementStat (mama_fid_t identifier)
- virtual void setName (const char *name)
- virtual void setLog (int log)
- virtual void setPublish (int publish)
- virtual void destroy (void)
- virtual mamaStatsCollector getStatsCollector ()

Protected Member Functions

- MamaStatsCollector (MamaStatsCollectorImpl *)

Protected Attributes

- MamaStatsCollectorImpl * mSimpl

7.51.1 Constructor & Destructor Documentation

7.51.1.1 Wombat::MamaStatsCollector::MamaStatsCollector (void)

7.51.1.2 virtual Wombat::MamaStatsCollector::~MamaStatsCollector (void) [virtual]

7.51.1.3 Wombat::MamaStatsCollector::MamaStatsCollector (MamaStatsCollectorImpl *) [protected]

7.51.2 Member Function Documentation

7.51.2.1 virtual void Wombat::MamaStatsCollector::create (mamaStatsCollectorType *type*, const char * *name*, const char * *middleware*) [virtual]

7.51.2.2 virtual void Wombat::MamaStatsCollector::incrementStat (mama_fid_t *identifier*) [virtual]

7.51.2.3 virtual void Wombat::MamaStatsCollector::setName (const char * *name*) [virtual]

7.51.2.4 virtual void Wombat::MamaStatsCollector::setLog (int *log*) [virtual]

7.51.2.5 virtual void Wombat::MamaStatsCollector::setPublish (int *publish*) [virtual]

7.51.2.6 virtual void Wombat::MamaStatsCollector::destroy (void) [virtual]

7.51.2.7 virtual mamaStatsCollector Wombat::MamaStatsCollector::getStatsCollector () [virtual]

7.51.3 Member Data Documentation

7.51.3.1 MamaStatsCollectorImpl* Wombat::MamaStatsCollector::mSimpl [protected]

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

- MamaStatsCollector.h

7.52 Wombat::MamaStatus Class Reference

`#include <MamaStatus.h>`

Public Member Functions

- virtual ~MamaStatus (void)
- MamaStatus (mama_status status)
- virtual const char * toString (void) const
- mama_status getStatus () const
- bool operator== (mama_status const rhs_i)
- bool operator== (MamaStatus const rhs_i)
- bool operator!= (mama_status const rhs_i)
- bool operator!= (MamaStatus const rhs_i)
- bool operator< (mama_status const rhs_i)
- bool operator< (MamaStatus const rhs_i)
- bool operator> (mama_status const rhs_i)
- bool operator> (MamaStatus const rhs_i)
- bool operator<= (mama_status const rhs_i)
- bool operator<= (MamaStatus const rhs_i)
- bool operator>= (mama_status const rhs_i)
- bool operator>= (MamaStatus const rhs_i)

7.52.1 Constructor & Destructor Documentation

7.52.1.1 virtual Wombat::MamaStatus::~MamaStatus (void) [virtual]

```
33 {}
```

7.52.1.2 Wombat::MamaStatus::MamaStatus (mama_status *status*)

```
36             : mStatus (status)
37         {}
```

7.52.2 Member Function Documentation

7.52.2.1 virtual const char* Wombat::MamaStatus::toString (void) const [virtual]

7.52.2.2 mama_status Wombat::MamaStatus::getStatus () const

```
42         {
```

```
43              return mStatus;
44          }
```

7.52.2.3 bool Wombat::MamaStatus::operator== (mama_status const *rhs_i*)

```
47          {
48              return mStatus == rhs_i;
49          }
```

7.52.2.4 bool Wombat::MamaStatus::operator== (MamaStatus const *rhs_i*)

```
52          {
53              return mStatus == rhs_i.getStatus();
54          }
```

7.52.2.5 bool Wombat::MamaStatus::operator!= (mama_status const *rhs_i*)

```
58          {
59              return !(mStatus == rhs_i);
60          }
```

7.52.2.6 bool Wombat::MamaStatus::operator!= (MamaStatus const *rhs_i*)

```
63          {
64              return !(mStatus == rhs_i.getStatus());
65          }
```

7.52.2.7 bool Wombat::MamaStatus::operator< (mama_status const *rhs_i*)

```
68          {
69              return mStatus < rhs_i;
70          }
```

7.52.2.8 bool Wombat::MamaStatus::operator< (MamaStatus const *rhs_i*)

```
73          {
74              return mStatus < rhs_i.getStatus();
75          }
```

7.52.2.9 bool Wombat::MamaStatus::operator> (mama_status const *rhs_i*)

```
78          {
79             return mStatus > rhs_i;
80          }
```

7.52.2.10 bool Wombat::MamaStatus::operator> (MamaStatus const *rhs_i*)

```
83          {
84             return mStatus > rhs_i.getStatus();
85          }
```

7.52.2.11 bool Wombat::MamaStatus::operator<= (mama_status const *rhs_i*)

```
88          {
89             return !(mStatus > rhs_i);
90          }
```

7.52.2.12 bool Wombat::MamaStatus::operator<= (MamaStatus const *rhs_i*)

```
93          {
94             return !(mStatus > rhs_i.getStatus());
95          }
```

7.52.2.13 bool Wombat::MamaStatus::operator>= (mama_status const *rhs_i*)

```
98          {
99             return !(mStatus < rhs_i);
100         }
```

7.52.2.14 bool Wombat::MamaStatus::operator>= (MamaStatus const *rhs_i*)

```
103         {
104            return !(mStatus < rhs_i.getStatus());
105         }
```

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

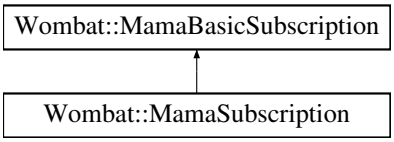
- MamaStatus.h

7.53 Wombat::MamaSubscription Class Reference

The MamaSubscription interface represents a subscription to a topic.

`#include <MamaSubscription.h>`

Inheritance diagram for Wombat::MamaSubscription::



Public Member Functions

- virtual ~MamaSubscription ()
- MamaSubscription (void)
- virtual void setCSubscription (mamaSubscription subscription)
- virtual mamaSubscription getCSubscription ()
- virtual void setup (MamaTransport *transport, MamaQueue *queue, MamaSubscriptionCallback *callback, const char *source, const char *symbol, void *closure=NULL)

  Set up a subscription.

- virtual void setup (MamaQueue *queue, MamaSubscriptionCallback *callback, MamaSource *source, const char *symbol, void *closure=NULL)

  Set up a subscription.

- virtual void activate ()

  Activate a subscriber using the throttle queue.

- virtual void deactivate ()

  Deactivate a subscriber.

- virtual void create (MamaTransport *transport, MamaQueue *queue, MamaSubscriptionCallback *callback, const char *source, const char *symbol, void *closure=NULL)

  Set up and activate a subscriber using the throttle queue.

- virtual void create (MamaQueue *queue, MamaSubscriptionCallback *callback, MamaSource *source, const char *symbol, void *closure=NULL)

  Set up and activate a subscriber using the throttle queue.

- virtual void createSnapshot (MamaTransport *transport, MamaQueue *queue, MamaSubscriptionCallback *callback, const char *source, const char *symbol, void *closure=NULL)

  Set up and activate a snapshot subscriber using the throttle queue.

- virtual void setRequiresInitial (bool requiresInitial)

  Determines whether the subscription requires an initial value.

- virtual bool getRequiresInitial (void)

  Return true if the subscription requires an initial value.

- virtual bool getReceivedInitial (void)

  Return true if the subscription has received an initial.

- virtual void setRetries (int retries)

  Set the number of retries for initial value requests and recap requests.

- virtual int getRetries (void)

  Return the retries.

- virtual void setSubscriptionType (mamaSubscriptionType type)

  Set the subscription type.

- virtual mamaSubscriptionType getSubscriptionType (void)

  Return the subscription type.

- virtual long getServiceLevelOpt (void)

  Return the service level option.

- virtual void setServiceLevel (mamaServiceLevel svcLevel, long svcLevelOpt)

  Set the service level.

- virtual mamaServiceLevel getServiceLevel (void)

  Return the service level.

- virtual const char * getSymbol (void) const

  Return the symbol for this subscription.

- virtual MamaSubscriptionCallback * getCallback (void) const
- virtual void setSymbol (const char *symbol)

  Sets the symbol for this subscription.

- virtual const MamaSourceDerivative * getSourceDerivative (void) const

  Return the (subscription-specific) MAMA source derivative for this subscription.

- virtual MamaSourceDerivative * getSourceDerivative (void)

  Return the (subscription-specific) MAMA source derivative for this subscription.

- virtual const MamaSource * getSource (void) const

  Return the MAMA source for this subscription.

- virtual const char * getSubscSource (void) const

  Return the source for this subscription.

- virtual void setTimeout (double timeout)

  Set the timeout for this subscription.

- virtual double getTimeout (void)

  Return the timeout.

- virtual void setRecoverGaps (bool recover)

  Attempt to recover from sequence number gaps by requesting a recap.

- virtual bool getRecoverGaps (void) const

  Returns true if listener is configure to recover from sequence number gaps by requesting a recap.

- virtual void setAppDataType (uint8_t dataType)

  Set the application data type.

- virtual uint8_t getAppDataType () const

  Get the application data type.

- virtual void setGroupSizeHint (int groupSizeHint)

  Set a hint to the size of groups when making group subscriptions.

- virtual void setItemClosure (void *closure)

  Set the item closure for group subscriptions.

- virtual void * getItemClosure (void)

  Get the item closure for group subscriptions.

- virtual void setPreInitialCacheSize (int cacheSize)

  Set the number of messages to cache for each symbol before the initial value arrives.

- virtual int getPreInitialCacheSize (void)

  Return the initial value cache size.

- virtual void setMsgQualifierFilter (bool ignoreDefinitelyDuplicate, bool ignorePossiblyDuplicate, bool ignoreDefinitelyDelyaed, bool ignorePossiblyDelayed, bool ignoreOutOfSequence)

  Set a filter to discard messages.

- virtual void getMsgQualifierFilter (bool &ignoreDefinitelyDuplicate, bool &ignorePossiblyDuplicate, bool &ignoreDefinitelyDelyaed, bool &ignorePossiblyDelayed, bool &ignoreOutOfSequence) const

  Get the filters that discard message according to the message qualifier.

- virtual void destroy ()

  Destroy the subscription.

- virtual void destroyEx ()

  This function will destroy the subscription and can be called from any thread.

7.53.1 Detailed Description

The `MamaSubscription` interface represents a subscription to a topic.

It provides transparent market data semantics and functionality including initial value requests, recap requests, subscription management and data quality.

7.53.2 Constructor & Destructor Documentation

7.53.2.1 virtual Wombat::MamaSubscription::~MamaSubscription () [virtual]

7.53.2.2 Wombat::MamaSubscription::MamaSubscription (void)

7.53.3 Member Function Documentation

7.53.3.1 virtual void Wombat::MamaSubscription::setCSubscription (mamaSubscription *subscription*) [virtual]

7.53.3.2 virtual mamaSubscription Wombat::MamaSubscription::getCSubscription () [virtual]

7.53.3.3 virtual void Wombat::MamaSubscription::setup (MamaTransport * *transport*, MamaQueue * *queue*, MamaSubscriptionCallback * *callback*, const char * *source*, const char * *symbol*, void * *closure* = NULL) [virtual]

Set up a subscription.

Parameters:

callback The callback.

transport The transport.

queue The mama queue.

source The data source name for the listener.

symbol The symbol for the listener.

closure The caller supplied closure.

7.53.3.4 virtual void Wombat::MamaSubscription::setup (MamaQueue * *queue*, MamaSubscriptionCallback * *callback*, MamaSource * *source*, const char * *symbol*, void * *closure* = NULL) [virtual]

Set up a subscription.

Parameters:

queue The mama queue.

callback The callback.

source The MamaSource identifying the publisher for this symbol.

symbol The symbol for the listener.

closure The caller supplied closure.

7.53.3.5 virtual void Wombat::MamaSubscription::activate () [virtual]

Activate a subscriber using the throttle queue.

This method places a request to create a subscriber on the throttle queue which dispatches tasks that produce messages at a controlled rate. The rate is determined by the outbound throttle rate of the underlying MamaTransport.

In the event that listener creation fails as the result of an messaging related error the callback is invoked with information regarding the error.

If entitlements are enabled, and the caller is not entitled to the requested symbol, the first invocation of the callback will invoked with status "MamaMsgStatus.NOT_-ENTITLED".

As an added convenience, callers may implement the onComplete and onError members of MamaSubscriptionCallback. onComplete is invoked prior to the arrival of any initial message signalling the successful creation of the listener. onError is invoked if a TIBRV or entitlement error occurs prior to listener creation.

It is also possible for an entitlement error to occur after a listener is created. This occurs when the entitlement information is included in the initial message sent by the feed handler as is often the case.

If an error occurs during listener creation. destroy is called automatically.

7.53.3.6 virtual void Wombat::MamaSubscription::deactivate () [virtual]

Deactivate a subscriber.

The subscription can be reactivated using activate().

7.53.3.7 virtual void Wombat::MamaSubscription::create (MamaTransport * *transport*, MamaQueue * *queue*, MamaSubscriptionCallback * *callback*, const char * *source*, const char * *symbol*, void * *closure* = NULL) [virtual]

Set up and activate a subscriber using the throttle queue.

This method is equivalent to calling setup() followed by activate().

Parameters:

transport The transport.

queue The mama queue.

callback The callback.

source The data source name for the listener.

symbol The symbol for the listener.

closure The caller supplied closure.

7.53.3.8 virtual void Wombat::MamaSubscription::create (MamaQueue * *queue*, MamaSubscriptionCallback * *callback*, MamaSource * *source*, const char * *symbol*, void * *closure* = NULL) [virtual]

Set up and activate a subscriber using the throttle queue.

This method is equivalent to calling setup() followed by activate().

Parameters:

queue The mama queue.

callback The callback.

source The MamaSource identifying the publisher for this symbol.

symbol The symbol for the listener.

closure The caller supplied closure.

7.53.3.9 virtual void Wombat::MamaSubscription::createSnapshot (MamaTransport * *transport*, MamaQueue * *queue*, MamaSubscriptionCallback * *callback*, const char * *source*, const char * *symbol*, void * *closure* = NULL) [virtual]

Set up and activate a snapshot subscriber using the throttle queue.

This method is equivalent to calling setup() followed by setServiceLevel(MAMA_SERVICE_LEVEL_SNAPSHOT,0) followed by activate().

Parameters:

transport The transport.

queue The mama queue.

callback The callback.

source The data source name for the listener.

symbol The symbol for the listener.

closure The caller supplied closure.

7.53.3.10 virtual void Wombat::MamaSubscription::setRequiresInitial (bool *requiresInitial*) [virtual]

Determines whether the subscription requires an initial value.

Must be set before calling createXXX(). Default is true. Not applicable for snapshot subscriptions as they simply request an initial value.

Parameters:

requiresInitial True if an initial value is required

7.53.3.11 virtual bool Wombat::MamaSubscription::getRequiresInitial (void) [virtual]

Return true if the subscription requires an initial value.

7.53.3.12 virtual bool Wombat::MamaSubscription::getReceivedInitial (void) [virtual]

Return true if the subscription has received an initial.

7.53.3.13 virtual void Wombat::MamaSubscription::setRetries (int *retries*) [virtual]

Set the number of retries for initial value requests and recap requests.

This must called before createXXX() to affect the initial value requests. Calling it after createXXX() only affects recap requests. The default is MAMA_DEFAULT_-RETRIES.

Parameters:

retries The number of time to retry the initial value request.

7.53.3.14 virtual int Wombat::MamaSubscription::getRetries (void) [virtual]

Return the retries.

7.53.3.15 virtual void Wombat::MamaSubscription::setSubscriptionType (mamaSubscriptionType *type*) [virtual]

Set the subscription type.

The default is normal.

Parameters:

type The type of subscription (normal, group, order book, etc.).

7.53.3.16 virtual mamaSubscriptionType Wombat::MamaSubscription::getSubscriptionType (void) [virtual]

Return the subscription type.

7.53.3.17 virtual long Wombat::MamaSubscription::getServiceLevelOpt (void) [virtual]

Return the service level option.

7.53.3.18 virtual void Wombat::MamaSubscription::setServiceLevel (mamaServiceLevel *svcLevel*, long *svcLevelOpt*) [virtual]

Set the service level.

This method must be invoked before createXXX().

Parameters:

svcLevel The service level of the subscription (real time, snapshot, etc.). The default is real time.

svcLevelOpt An optional argument for certain service levels.

7.53.3.19 virtual mamaServiceLevel Wombat::MamaSubscription::getServiceLevel (void) [virtual]

Return the service level.

7.53.3.20 virtual const char* Wombat::MamaSubscription::getSymbol (void) const [virtual]

Return the symbol for this subscription.

Returns:

The topic.

7.53.3.21 virtual MamaSubscriptionCallback* Wombat::MamaSubscription::getCallback (void) const [virtual]

7.53.3.22 virtual void Wombat::MamaSubscription::setSymbol (const char * *symbol*) [virtual]

Sets the symbol for this subscription.

Should generally only be used for updating symbology mappings.

7.53.3.23 virtual const MamaSourceDerivative* Wombat::MamaSubscription::getSourceDerivative (void) const [virtual]

Return the (subscription-specific) MAMA source derivative for this subscription.

Returns:

The source derivative.

7.53.3.24 virtual MamaSourceDerivative* Wombat::MamaSubscription::getSourceDerivative (void) [virtual]

Return the (subscription-specific) MAMA source derivative for this subscription.

Returns:

The source derivative.

7.53.3.25 virtual const MamaSource* Wombat::MamaSubscription::getSource (void) const [virtual]

Return the MAMA source for this subscription.

Returns:

The source.

7.53.3.26 virtual const char* Wombat::MamaSubscription::getSubscSource (void) const [virtual]

Return the source for this subscription.

Returns:

The source.

7.53.3.27 virtual void Wombat::MamaSubscription::setTimeout (double *timeout*) [virtual]

Set the timeout for this subscription.

The timeout is used for requesting initial values, and recaps.

Parameters:

timeout The timeout in seconds.

7.53.3.28 virtual double Wombat::MamaSubscription::getTimeout (void) [virtual]

Return the timeout.

7.53.3.29 virtual void Wombat::MamaSubscription::setRecoverGaps (bool *recover*) [virtual]

Attempt to recover from sequence number gaps by requesting a recap.

Parameters:

recover true enables recovery attempts.

7.53.3.30 virtual bool Wombat::MamaSubscription::getRecoverGaps (void) const [virtual]

Returns true if listener is configure to recover from sequence number gaps by requesting a recap.

Returns:

true if gap recover is enabled.

7.53.3.31 virtual void Wombat::MamaSubscription::setAppDataType (uint8_t *dataType*) [virtual]

Set the application data type.

The default is 0.

Parameters:

dataType The application-specific data type (e.g., market data).

7.53.3.32 virtual uint8_t Wombat::MamaSubscription::getAppDataType () const [virtual]

Get the application data type.

The default is 0.

Returns:

The application-specific data type (e.g., market data).

7.53.3.33 virtual void Wombat::MamaSubscription::setGroupSizeHint (int *groupSizeHint*) [virtual]

Set a hint to the size of groups when making group subscriptions.

Parameters:

groupSizeHint Approximate expected group size

7.53.3.34 virtual void Wombat::MamaSubscription::setItemClosure (void * *closure*) [virtual]

Set the item closure for group subscriptions.

Group subscriptions receive updates for multiple symbols. This method allows calls to set a per-symbol closure which will be passed as the fourth argument to subsequent calls to the onMsg callback. This method may only be called during the onMsg callback.

Setting the item closure for a non-group subscription provides a second closure.

7.53.3.35 virtual void* Wombat::MamaSubscription::getItemClosure (void) [virtual]

Get the item closure for group subscriptions.

See setItemClosure. When invoked during an onMsg callback this method returns the closure for the current item in a group subscription. When invoked outside an onMsg callback, it returns the closure from the most recent callback.

7.53.3.36 virtual void Wombat::MamaSubscription::setPreInitialCacheSize (int *cacheSize*) [virtual]

Set the number of messages to cache for each symbol before the initial value arrives.

This allows the subscription to recover when the initial value arrives late (after a subsequent trade or quote already arrived).

For group subscription, a separate cache is used for each group member.

The default is 10.

Parameters:

cacheSize The size of the cache.

7.53.3.37 virtual int Wombat::MamaSubscription::getPreInitialCacheSize (void) [virtual]

Return the initial value cache size.

Returns:

The cache size.

7.53.3.38 virtual void Wombat::MamaSubscription::setMsgQualifierFilter (bool *ignoreDefinitelyDuplicate*, bool *ignorePossiblyDuplicate*, bool *ignoreDefinitelyDelyaed*, bool *ignorePossiblyDelayed*, bool *ignoreOutOfSequence*) [virtual]

Set a filter to discard messages.

Parameters:

ignoreDefinitelyDuplicate If true callbacks will not be invoked for messages where MamaMsg::getIsDefinitelyDuplicate returns true.

ignorePossiblyDuplicate If true callbacks will not be invoked for messages where MamaMsg::getIsPossiblyDuplicate returns true.

ignoreDefinitelyDelyaed If true callbacks will not be invoked for messages where MamaMsg::getIsDefinitelyDelayed returns true.

ignorePossiblyDelayed If true callbacks will not be invoked for messages where MamaMsg::getIsPossiblyDelayed returns true.

ignoreOutOfSequence If true callbacks will not be invoked for messages where MamaMsg::getIsOutOfSequence returns true.

7.53.3.39 virtual void Wombat::MamaSubscription::getMsgQualifierFilter (bool & *ignoreDefinitelyDuplicate*, bool & *ignorePossiblyDuplicate*, bool & *ignoreDefinitelyDelyaed*, bool & *ignorePossiblyDelayed*, bool & *ignoreOutOfSequence*) const [virtual]

Get the filters that discard message according to the message qualifier.

Parameters:

ignoreDefinitelyDuplicate If true callbacks will not be invoked for messages where MamaMsg::getIsDefinitelyDuplicate returns true.

ignorePossiblyDuplicate If true callbacks will not be invoked for messages where MamaMsg::getIsPossiblyDuplicate returns true.

ignoreDefinitelyDelayed If true callbacks will not be invoked for messages where MamaMsg::getIsDefinitelyDelayed returns true.

ignorePossiblyDelayed If true callbacks will not be invoked for messages where MamaMsg::getIsPossiblyDelayed returns true.

ignoreOutOfSequence If true callbacks will not be invoked for messages where MamaMsg::getIsOutOfSequence returns true.

7.53.3.40 virtual void Wombat::MamaSubscription::destroy () [virtual]

Destroy the subscription.

Destroys the underlying subscription. The subscription can be recreated via a subsequent call to create()

Reimplemented from Wombat::MamaBasicSubscription.

7.53.3.41 virtual void Wombat::MamaSubscription::destroyEx () [virtual]

This function will destroy the subscription and can be called from any thread.

Note that the subscription will not be fully destroyed until the onDestroy callback is received from the MamaBasicSubscriptionCallback interface. To destroy from the dispatching thread the destroy function should be used in preference.

Reimplemented from Wombat::MamaBasicSubscription.

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

- MamaSubscription.h

7.54 Wombat::MamaSubscriptionCallback Class Reference

The message callback interface.

`#include <MamaSubscriptionCallback.h>`

Public Member Functions

- virtual ~MamaSubscriptionCallback ()
- virtual void onCreate (MamaSubscription *subscription)=0

  Method invoked when subscription creation is complete, and before any calls to `onMsg`.

- virtual void onError (MamaSubscription *subscription, const MamaStatus &status, const char *symbol)=0

  Invoked if an error occurs during prior to subscription creation or if the subscription receives a message for an unentitled symbol.

- virtual void onGap (MamaSubscription *subscription)

  Method invoked when a sequence number gap is detected.

- virtual void onDestroy (MamaSubscription *subscription)

  Method invoked when a subscription has been destroyed through destroyEx.

- virtual void onRecapRequest (MamaSubscription *subscription)

  Method invoked when a recap is requested upon detecting a sequence number gap.

- virtual void onMsg (MamaSubscription *subscription, MamaMsg &msg)=0

  Invoked when a message arrives.

- virtual void onQuality (MamaSubscription *subscription, mamaQuality quality, const char *symbol, short cause, const void *platformInfo)=0

  Invoked when the quality of this subscription changes.

- virtual void onCreate (MamaBasicSubscription *subscription)
- virtual void onError (MamaBasicSubscription *subscription, const MamaStatus &status, const char *symbol)
- virtual void onMsg (MamaBasicSubscription *subscription, MamaMsg &msg)

  Invoked when a message arrives.

7.54.1 Detailed Description

The message callback interface.

Callers provide an object implementing this interface on creating a MamaSubscription.

See also:

MamaSubscription

Author:

mls

7.54.2 Constructor & Destructor Documentation

7.54.2.1 virtual Wombat::MamaSubscriptionCallback::~MamaSubscriptionCallback () [virtual]

```
43         {};
```

7.54.3 Member Function Documentation

7.54.3.1 virtual void Wombat::MamaSubscriptionCallback::onCreate (MamaSubscription * *subscription*) [pure virtual]

Method invoked when subscription creation is complete, and before any calls to onMsg.

Since subscriptions are created asynchronous by throttle, this callback provides the subscription instance after the throttle processes the creation request.

Parameters:

subscription The subscription.

7.54.3.2 virtual void Wombat::MamaSubscriptionCallback::onError (MamaSubscription * *subscription*, const MamaStatus & *status*, const char * *symbol*) [pure virtual]

Invoked if an error occurs during prior to subscription creation or if the subscription receives a message for an unentitled symbol.

If the status is MamaMsgStatus.NOT_ENTITTLED the symbol parameter is the specific unentitled symbol. If the subscription symbol contains wildcards, the subscription may still receive messages for other entitled symbol.

Parameters:

subscription The subscription.

status The wombat error code.

symbol The symbol for NOT_ENTITLED

7.54.3.3 virtual void Wombat::MamaSubscriptionCallback::onGap (MamaSubscription * *subscription*) [virtual]

Method invoked when a sequence number gap is detected.

At this point the topic is considered stale and the subscription will not receive further messages until the feed handler satisfies a recap request.

Parameters:

subscription The subscription.

```
83         {};
```

7.54.3.4 virtual void Wombat::MamaSubscriptionCallback::onDestroy (MamaSubscription * *subscription*) [virtual]

Method invoked when a subscription has been destroyed through destroyEx.

Parameters:

subscription The subscription.

```
92         {};
```

7.54.3.5 virtual void Wombat::MamaSubscriptionCallback::onRecapRequest (MamaSubscription * *subscription*) [virtual]

Method invoked when a recap is requested upon detecting a sequence number gap.

Parameters:

subscription The subscription.

```
101         {};
```

7.54.3.6 virtual void Wombat::MamaSubscriptionCallback::onMsg (MamaSubscription * *subscription*, MamaMsg & *msg*) [pure virtual]

Invoked when a message arrives.

Parameters:

subscription the MamaSubscription.

msg The MamaMsg which resulted in this callback being invoked.

7.54.3.7 virtual void Wombat::MamaSubscriptionCallback::onQuality (MamaSubscription * *subscription*, mamaQuality *quality*, const char * *symbol*, short *cause*, const void * *platformInfo*) [pure virtual]

Invoked when the quality of this subscription changes.

Parameters:

subscription The subscription.

quality The new quality: one of the values in the MamaQuality class.

symbol The symbol for this subscription.

cause The cause of the quality event

platformInfo Info associated with the quality event

The cause and platformInfo are supplied only by some middlewares. The information provided by platformInfo is middleware specific. The following middlewares are supported:

tibrv: provides the char* version of the tibrv advisory message.

7.54.3.8 virtual void Wombat::MamaSubscriptionCallback::onCreate (MamaBasicSubscription * *subscription*) [virtual]

```
137         {
138             onCreate ((MamaSubscription*)subscription);
139         }
```

7.54.3.9 virtual void Wombat::MamaSubscriptionCallback::onError (MamaBasicSubscription * *subscription*, const MamaStatus & *status*, const char * *symbol*) [virtual]

```
144         {
145             onError ((MamaSubscription*)subscription, status, symbol);
146         }
```

7.54.3.10 virtual void Wombat::MamaSubscriptionCallback::onMsg (MamaBasicSubscription * *subscription*, MamaMsg & *msg*) [virtual]

Invoked when a message arrives.

Parameters:

subscription the MamaSubscription.

msg The TibrvMsg.

```
156         {
157             onMsg ((MamaSubscription*)subscription, msg);
158         }
```

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

- MamaSubscriptionCallback.h

7.55 Wombat::MamaSubscriptionIteratorCallback Class Reference

`#include <MamaSource.h>`

Public Member Functions

- virtual void onSubscription (MamaSource *source, MamaSubscription *subscription, void *closure)
- virtual ~MamaSubscriptionIteratorCallback ()

7.55.1 Constructor & Destructor Documentation

7.55.1.1 virtual Wombat::MamaSubscriptionIterator-Callback::~MamaSubscriptionIteratorCallback () [virtual]

```
47         {};
```

7.55.2 Member Function Documentation

7.55.2.1 virtual void Wombat::MamaSubscriptionIteratorCallback::onSubscription (MamaSource * *source*, MamaSubscription * *subscription*, void * *closure*) [virtual]

```
42         {
43             return;
44         }
```

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

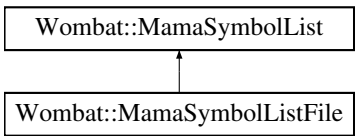
- MamaSource.h

7.56 Wombat::MamaSymbolList Class Reference

MamaSymbolList manages a list of MAMA symbols and related attributes.

`#include <MamaSymbolList.h>`

Inheritance diagram for Wombat::MamaSymbolList::



Public Member Functions

- MamaSymbolList ()
- virtual ~MamaSymbolList ()
- void addMembershipHandler (MamaSymbolListMembershipHandler *handler)

  Add a "membership" handler that implements the MamaSymbolListMembership-Handler interface.

- void addMember (MamaSymbolListMember *member)

  Add a symbol to the list.

- MamaSymbolListMember * findMember (const char *symbol, const char *source, mamaTransport transport)

  Find a symbol in the list.

- MamaSymbolListMember * removeMember (const char *symbol, const char *source, mamaTransport transport)

  Remove a symbol from the list (providing it exists in the list).

- void removeMember (MamaSymbolListMember &member)

  Remove a symbol from the list (providing it exists in the list).

- void removeMemberAll (void)

  Remove all symbols from the list (providing it exists in the list).

- void clear ()

  Remove all symbols from the list.

- void dump ()

  Dump the contents of the list to stdout.

- bool empty () const
- mama_size_t size () const
- void setClosure (void *closure)

  Set the closure.

- void * getClosure () const

  Get the closure.

- void iterate (MamaSymbolListIteratorHandler &handler, void *iterateClosure=NULL)

  Iterate over all members of the symbol list.

- mamaSymbolList getCValue ()

  Get the underlying Impl at C level.

- const mamaSymbolList getCValue () const

  Get the underlying Impl at C level.

Public Attributes

- MamaSymbolListImpl * myPimpl

Protected Attributes

- mamaSymbolList myList

7.56.1 Detailed Description

MamaSymbolList manages a list of MAMA symbols and related attributes.

Methods are provided for creating, updating and sorting the members of the list. Handler interfaces are provided so that it is possible to handle asynchronous/external changes to the symbol list, as many types of symbol lists can be quite dynamic.

7.56.2 Constructor & Destructor Documentation

7.56.2.1 Wombat::MamaSymbolList::MamaSymbolList ()

7.56.2.2 virtual Wombat::MamaSymbolList::~MamaSymbolList () [virtual]

7.56.3 Member Function Documentation

7.56.3.1 void Wombat::MamaSymbolList::addMembershipHandler (MamaSymbolListMembershipHandler * *handler*)

Add a "membership" handler that implements the MamaSymbolListMembership-Handler interface.

Multiple handlers may be registered.

Parameters:

handler The handler to be registered.

7.56.3.2 void Wombat::MamaSymbolList::addMember (MamaSymbolListMember * *member*)

Add a symbol to the list.

The list maintains a unique list of symbols.

Parameters:

member The symbol member to be added.

7.56.3.3 MamaSymbolListMember* Wombat::MamaSymbolList::findMember (const char * *symbol*, const char * *source*, mamaTransport *transport*)

Find a symbol in the list.

Parameters:

symbol The name of the symbol to be removed.

source The source of the symbol to be removed.

transport The transport of the symbol to be removed.

Returns:

The object containing additional information about the symbol (or NULL).

7.56.3.4 MamaSymbolListMember* Wombat::MamaSymbolList::removeMember (const char * *symbol*, const char * *source*, mamaTransport *transport*)

Remove a symbol from the list (providing it exists in the list).

The member itself is not destroyed but returned as the result of this method.

Parameters:

symbol The symbol to be removed.

source The source of the symbol to be removed

transport The transport of the symbol to be removed

Returns:

The member just removed (or NULL if not found)

7.56.3.5 void Wombat::MamaSymbolList::removeMember (MamaSymbolListMember & *member*)

Remove a symbol from the list (providing it exists in the list).

The member itself is not destroyed.

Parameters:

member The member to be removed.

7.56.3.6 void Wombat::MamaSymbolList::removeMemberAll (void)

Remove all symbols from the list (providing it exists in the list).

The member itself is not destroyed.

7.56.3.7 void Wombat::MamaSymbolList::clear ()

Remove all symbols from the list.

Handlers remain registered.

7.56.3.8 void Wombat::MamaSymbolList::dump ()

Dump the contents of the list to stdout.

For debugging.

7.56.3.9 bool Wombat::MamaSymbolList::empty () const

Returns:

whether the symbol list is empty.

7.56.3.10 mama_size_t Wombat::MamaSymbolList::size () const

Returns:

the size of the symbol list.

7.56.3.11 void Wombat::MamaSymbolList::setClosure (void * *closure*)

Set the closure.

Parameters:

closure The closure.

7.56.3.12 void* Wombat::MamaSymbolList::getClosure () const

Get the closure.

Returns:

The closure.

7.56.3.13 void Wombat::MamaSymbolList::iterate (MamaSymbolListIteratorHandler & *handler*, void * *iterateClosure* = NULL)

Iterate over all members of the symbol list.

Parameters:

handler Handler invoked for each member of the symbol list.

iterateClosure The closure passed to the MamaSymbolListIteratorHandler::onMember() interface.

7.56.3.14 mamaSymbolList Wombat::MamaSymbolList::getCValue ()

Get the underlying Impl at C level.

Returns:

The mamaSymbolList

7.56.3.15 const mamaSymbolList Wombat::MamaSymbolList::getCValue () const

Get the underlying Impl at C level.

Returns:

The mamaSymbolList

7.56.4 Member Data Documentation

7.56.4.1 MamaSymbolListImpl* Wombat::MamaSymbolList::myPimpl

7.56.4.2 mamaSymbolList Wombat::MamaSymbolList::myList [protected]

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

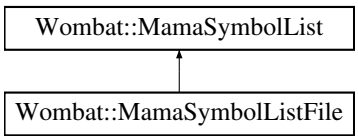
- MamaSymbolList.h

7.57 Wombat::MamaSymbolListFile Class Reference

MamaSymbolListFile is a file based symbol list with the ability to detect external changes to the file.

`#include <MamaSymbolListFile.h>`

Inheritance diagram for Wombat::MamaSymbolListFile::



Public Member Functions

- MamaSymbolListFile ()
- ~MamaSymbolListFile ()
- void setFileName (const char *fileName)

  Set the file name.

- void setBridge (const mamaBridge bridge)

  Set the bridge to use for creating transports.

- void readFile ()

  Read the contents of the file into the symbol list.

- void writeFile ()

  Write the symbol list to the file.

- void setFileMonitor (double checkSeconds, MamaQueue *queue)

  Set a timer to check whether the file has changed its contents.

7.57.1 Detailed Description

MamaSymbolListFile is a file based symbol list with the ability to detect external changes to the file.

The file's structure is as follows:

- One symbol per line.

- Whitespace-separated columns for the symbol, the source and the MAMA transport name.
- The source and transport names are optional; if the transport name is present then so must be the source name.

For example: IBM CTA default DIS CTA default MSFT NASDAQ default

7.57.2 Constructor & Destructor Documentation

7.57.2.1 Wombat::MamaSymbolListFile::MamaSymbolListFile ()

7.57.2.2 Wombat::MamaSymbolListFile::~MamaSymbolListFile ()

7.57.3 Member Function Documentation

7.57.3.1 void Wombat::MamaSymbolListFile::setFileName (const char * *fileName*)

Set the file name.

7.57.3.2 void Wombat::MamaSymbolListFile::setBridge (const mamaBridge *bridge*)

Set the bridge to use for creating transports.

7.57.3.3 void Wombat::MamaSymbolListFile::readFile ()

Read the contents of the file into the symbol list.

7.57.3.4 void Wombat::MamaSymbolListFile::writeFile ()

Write the symbol list to the file.

7.57.3.5 void Wombat::MamaSymbolListFile::setFileMonitor (double *checkSeconds*, MamaQueue * *queue*)

Set a timer to check whether the file has changed its contents.

If the contents have changed, various add/remove membership events may be applied to the MamaSymbolList.

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

- MamaSymbolListFile.h

7.58 Wombat::MamaSymbolListIteratorHandler Class Reference

The MamaSymbolListIteratorHandler interface.

`#include <MamaSymbolListHandlerTypes.h>`

Public Member Functions

- virtual ~MamaSymbolListIteratorHandler ()
- virtual void onMember (MamaSymbolList &symbolList, MamaSymbolListMember &member, void *iterateClosure)=0

  Method invoked for every member in the symbol list.

- virtual void onComplete (MamaSymbolList &symbolList, void *iterateClosure)

  Method invoked after all symbol list members have been iterated over.

7.58.1 Detailed Description

The MamaSymbolListIteratorHandler interface.

Instances of subclasses of this class can be passed to the MamaSymbolList::iterate() method and the onMember() interface will be invoked for each member in the list.

7.58.2 Constructor & Destructor Documentation

7.58.2.1 virtual Wombat::MamaSymbolListIteratorHandler::~MamaSymbolListIteratorHandler () [virtual]

```
41 {};
```

7.58.3 Member Function Documentation

7.58.3.1 virtual void Wombat::MamaSymbolListIteratorHandler::onMember (MamaSymbolList & *symbolList*, MamaSymbolListMember & *member*, void * *iterateClosure*) [pure virtual]

Method invoked for every member in the symbol list.

Parameters:

symbolList The symbol list.

member The member of the symbol list.

iterateClosure The closure argument to MamaSymbolList::iterate().

7.58.3.2 virtual void Wombat::MamaSymbolListIteratorHandler::onComplete (MamaSymbolList & *symbolList*, void * *iterateClosure*) [virtual]

Method invoked after all symbol list members have been iterated over.

Parameters:

symbolList The symbol list.

iterateClosure The closure argument to MamaSymbolList::iterate().

```
61                                                                    {}
```

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

- MamaSymbolListHandlerTypes.h

7.59 Wombat::MamaSymbolListMember Class Reference

The MamaSymbolListMember class represents the information about the symbol list member.

`#include <MamaSymbolListMember.h>`

Public Member Functions

- MamaSymbolListMember (MamaSymbolList *symbolList)

  Construct a symbol list member that can be added to the symbol list.

- MamaSymbolListMember (MamaSymbolList *list, mamaSymbolListMember cMember)

  Construct symbol list member that wraps the given C symbol list member.

- virtual ~MamaSymbolListMember ()
- const char * getSymbol () const

  Get the symbol name.

- const char * getSource () const

  Get the source name.

- MamaTransport * getTransport () const

  Get the MAMA transport.

- MamaSymbolList * getSymbolList () const

  Get the MAMA symbol list to which this member belongs.

- void * getClosure () const

  Get the closure argument.

- void setClosure (void *closure)

  Set the closure argument.

- void setSymbol (const char *symbol)

  Set the symbol name.

- void setSource (const char *source)

  Set the source name.

- void setTransport (MamaTransport *transport)

  Set the MAMA transport.

- void setSymbolList (MamaSymbolList *symbolList)

  Set the MAMA symbol list to which this member belongs.

Protected Member Functions

- mamaSymbolListMember getCimpl ()

  Get a handle to the underlying C implementation.

Friends

- class MamaSymbolList

7.59.1 Detailed Description

The MamaSymbolListMember class represents the information about the symbol list member.

In addition to the symbols, it is also possible to obtain information about the "source" of the symbol and which MamaTransport it can be found on.

7.59.2 Constructor & Destructor Documentation

7.59.2.1 Wombat::MamaSymbolListMember::MamaSymbolListMember (MamaSymbolList * *symbolList*)

Construct a symbol list member that can be added to the symbol list.

7.59.2.2 Wombat::MamaSymbolListMember::MamaSymbolListMember (MamaSymbolList * *list*, mamaSymbolListMember *cMember*)

Construct symbol list member that wraps the given C symbol list member.

7.59.2.3 virtual Wombat::MamaSymbolListMember::~MamaSymbolListMember () [virtual]

7.59.3 Member Function Documentation

7.59.3.1 const char* Wombat::MamaSymbolListMember::getSymbol () const

Get the symbol name.

7.59.3.2 const char* Wombat::MamaSymbolListMember::getSource () const

Get the source name.

7.59.3.3 MamaTransport* Wombat::MamaSymbolListMember::getTransport () const

Get the MAMA transport.

7.59.3.4 MamaSymbolList* Wombat::MamaSymbolListMember::getSymbolList () const

Get the MAMA symbol list to which this member belongs.

7.59.3.5 void* Wombat::MamaSymbolListMember::getClosure () const

Get the closure argument.

7.59.3.6 void Wombat::MamaSymbolListMember::setClosure (void * *closure*)

Set the closure argument.

7.59.3.7 void Wombat::MamaSymbolListMember::setSymbol (const char * *symbol*)

Set the symbol name.

7.59.3.8 void Wombat::MamaSymbolListMember::setSource (const char * *source*)

Set the source name.

7.59.3.9 void Wombat::MamaSymbolListMember::setTransport (MamaTransport * *transport*)

Set the MAMA transport.

7.59.3.10 void Wombat::MamaSymbolListMember::setSymbolList (MamaSymbolList * *symbolList*)

Set the MAMA symbol list to which this member belongs.

7.59.3.11 mamaSymbolListMember Wombat::MamaSymbolListMember::getCimpl () [protected]

Get a handle to the underlying C implementation.

7.59.4 Friends And Related Function Documentation

7.59.4.1 friend class MamaSymbolList [friend]

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

- MamaSymbolListMember.h

7.60 Wombat::MamaSymbolListMembershipHandler Class Reference

The MamaSymbolListMembershipHandler interface.

`#include <MamaSymbolListHandlerTypes.h>`

Public Member Functions

- virtual ~MamaSymbolListMembershipHandler ()
- virtual void onMemberAdd (MamaSymbolList &symbolList, MamaSymbolListMember &member)=0

  Method invoked when a symbol has been added to the list.

- virtual void onMemberRemove (MamaSymbolList &symbolList, MamaSymbolListMember &member)=0

  Method invoked immediately prior to a symbol being removed from the list.

- virtual void onOrderChange (MamaSymbolList &symbolList)=0

  Method invoked when the sorting of the symbol list has changed.

7.60.1 Detailed Description

The MamaSymbolListMembershipHandler interface.

Instances of subclasses of this class can be registered with a MamaSymbolList in order to be notified of symbol add/remove events to the symbol list.

7.60.2 Constructor & Destructor Documentation

7.60.2.1 virtual Wombat::MamaSymbolListMembershipHandler::~MamaSymbolListMembershipHandler () [virtual]

`74 {};`

7.60.3 Member Function Documentation

7.60.3.1 virtual void Wombat::MamaSymbolListMembership-Handler::onMemberAdd (MamaSymbolList & *symbolList*, MamaSymbolListMember & *member*) [pure virtual]

Method invoked when a symbol has been added to the list.

Invocation of this function is conditional and is subject to a positive return (true) from the prior invocation of `onDeclareInterest` where the registered object has the opportunity to declare their interest in subsequent events on the given symbol.

Parameters:

symbolList The symbolList.

member The member just added to the list.

7.60.3.2 virtual void Wombat::MamaSymbolListMembership-Handler::onMemberRemove (MamaSymbolList & *symbolList*, MamaSymbolListMember & *member*) [pure virtual]

Method invoked immediately prior to a symbol being removed from the list.

Invocation of this function is conditional and is subject to a positive return (true) from the prior invocation of `onDeclareInterest` where the registered object has the opportunity to declare their interest in subsequent events on the given symbol.

Parameters:

symbolList The symbolList.

member The member just removed from the list.

7.60.3.3 virtual void Wombat::MamaSymbolListMembershipHandler::on-OrderChange (MamaSymbolList & *symbolList*) [pure virtual]

Method invoked when the sorting of the symbol list has changed.

Parameters:

symbolList The symbol list.

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

- MamaSymbolListHandlerTypes.h

7.61 Wombat::MamaSymbolListResortHandler Class Reference

The MamaSymbolListResortHandler interface.

`#include <MamaSymbolListHandlerTypes.h>`

Public Member Functions

- virtual ~MamaSymbolListResortHandler ()
- virtual void onResort (MamaSymbolList *symbolList)=0

  Method invoked when the sorting of the symbol list has changed.

7.61.1 Detailed Description

The MamaSymbolListResortHandler interface.

Instances of subclasses of this class can be registered with a MamaSymbolList and will be invoked when the sorting of the symbol list has changed.

7.61.2 Constructor & Destructor Documentation

7.61.2.1 virtual Wombat::MamaSymbolListResortHandler::~MamaSymbolListResortHandler () [virtual]

```
121 {};
```

7.61.3 Member Function Documentation

7.61.3.1 virtual void Wombat::MamaSymbolListResortHandler::onResort (MamaSymbolList * *symbolList*) [pure virtual]

Method invoked when the sorting of the symbol list has changed.

Parameters:

symbolList The symbol list.

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

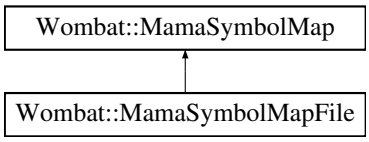
- MamaSymbolListHandlerTypes.h

7.62 Wombat::MamaSymbolMap Class Reference

The MamaSymbolMap class provides the ability for MAMA to do client side symbology mapping.

`#include <MamaSymbolMap.h>`

Inheritance diagram for Wombat::MamaSymbolMap::



Public Member Functions

- virtual ~MamaSymbolMap (void)
- virtual bool map (char *result, const char *symbol, size_t maxLen) const =0

  Map a symbol.

- virtual bool revMap (char *result, const char *symbol, size_t maxLen) const =0

  Map a feed side symbol.

7.62.1 Detailed Description

The MamaSymbolMap class provides the ability for MAMA to do client side symbology mapping.

Subclasses of this class can provide custom symbology mapping. A MamaSymbolMap can be assigned to each MamaTransport.

7.62.2 Constructor & Destructor Documentation

7.62.2.1 virtual Wombat::MamaSymbolMap::~MamaSymbolMap (void) [virtual]

```
39 {}
```

7.62.3 Member Function Documentation

7.62.3.1 virtual bool Wombat::MamaSymbolMap::map (char * *result*, const char * *symbol*, size_t *maxLen*) const [pure virtual]

Map a symbol.

The result is the feed side symbol used to actually subscribe to in the infrastructure. The return value indicates whether or not a symbology mapping existed for the given symbol (True = yes, False = No).

Implemented in Wombat::MamaSymbolMapFile.

7.62.3.2 virtual bool Wombat::MamaSymbolMap::revMap (char * *result*, const char * *symbol*, size_t *maxLen*) const [pure virtual]

Map a feed side symbol.

This is reverse of the natural mapping schema and the result in this case is the corresponding client side symbol. The return value indicates whether or not a symbology mapping existed for the given symbol (True = yes, False = No).

Implemented in Wombat::MamaSymbolMapFile.

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

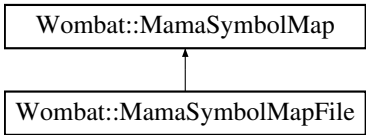
- MamaSymbolMap.h

7.63 Wombat::MamaSymbolMapFile Class Reference

MamaSymbolMapFile is a concrete implementation of a symbology map.

`#include <MamaSymbolMapFile.h>`

Inheritance diagram for Wombat::MamaSymbolMapFile::



Public Member Functions

- MamaSymbolMapFile ()
- virtual ~MamaSymbolMapFile ()
- virtual mama_status load (const char *mapFileName)
- virtual void addMap (const char *fromSymbol, const char *toSymbol)
- virtual bool map (char *result, const char *symbol, size_t maxLen) const

  Map a symbol.

- virtual bool revMap (char *result, const char *symbol, size_t maxLen) const

  Map a feed side symbol.

7.63.1 Detailed Description

MamaSymbolMapFile is a concrete implementation of a symbology map.

It can load a filename and expects the contents of that file to contain two columns of data, with the columns separated by white space. A matching symbol of the left column is mapped to the symbol in the right column. If the symbol does not match anything in the file (or the file cannot be found), then the original symbol is used (no mapping).

7.63.2 Constructor & Destructor Documentation

7.63.2.1 Wombat::MamaSymbolMapFile::MamaSymbolMapFile ()

7.63.2.2 virtual Wombat::MamaSymbolMapFile::~MamaSymbolMapFile () [virtual]

7.63.3 Member Function Documentation

7.63.3.1 virtual mama_status Wombat::MamaSymbolMapFile::load (const char * *mapFileName*) [virtual]

7.63.3.2 virtual void Wombat::MamaSymbolMapFile::addMap (const char * *fromSymbol*, const char * *toSymbol*) [virtual]

7.63.3.3 virtual bool Wombat::MamaSymbolMapFile::map (char * *result*, const char * *symbol*, size_t *maxLen*) const [virtual]

Map a symbol.

The result is the feed side symbol used to actually subscribe to in the infrastructure. The return value indicates whether or not a symbology mapping existed for the given symbol (True = yes, False = No).

Implements Wombat::MamaSymbolMap.

7.63.3.4 virtual bool Wombat::MamaSymbolMapFile::revMap (char * *result*, const char * *symbol*, size_t *maxLen*) const [virtual]

Map a feed side symbol.

This is reverse of the natural mapping schema and the result in this case is the corresponding client side symbol. The return value indicates whether or not a symbology mapping existed for the given symbol (True = yes, False = No).

Implements Wombat::MamaSymbolMap.

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

- MamaSymbolMapFile.h

7.64 Wombat::MamaSymbolSource Class Reference

MamaSymbolSource defines an interface which all SymbolSources should implement in order to provide a mechanism by which objects implementing the "MamaSymbolSourceCallback::onSymbol" can be registered with the source such that they can be notified of new symbols as they arrive.

`#include <MamaSymbolSource.h>`

Public Member Functions

- virtual ~MamaSymbolSource ()
- virtual void addHandler (MamaSymbolSourceCallback *callback)=0

7.64.1 Detailed Description

MamaSymbolSource defines an interface which all SymbolSources should implement in order to provide a mechanism by which objects implementing the "MamaSymbolSourceCallback::onSymbol" can be registered with the source such that they can be notified of new symbols as they arrive.

7.64.2 Constructor & Destructor Documentation

7.64.2.1 virtual Wombat::MamaSymbolSource::~MamaSymbolSource () [virtual]

```
40 {};
```

7.64.3 Member Function Documentation

7.64.3.1 virtual void Wombat::MamaSymbolSource::addHandler (MamaSymbolSourceCallback * *callback*) [pure virtual]

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

- MamaSymbolSource.h

7.65 Wombat::MamaSymbolSourceCallback Class Reference

The MamaSymbolSourceCallback interface.

`#include <MamaSymbolSourceCallback.h>`

Public Member Functions

- virtual ~MamaSymbolSourceCallback ()
- virtual void onSymbol (MamaSymbolSource *symbolSource, const char *symbol, void *closure)=0

  Method invoked when a symbol arrives at a symbol source.

7.65.1 Detailed Description

The MamaSymbolSourceCallback interface.

The single callback function "onSymbol" gets invoked when a symbol arrives at a symbol source.

7.65.2 Constructor & Destructor Documentation

7.65.2.1 virtual Wombat::MamaSymbolSourceCallback::~MamaSymbolSourceCallback () [virtual]

```
38 {};
```

7.65.3 Member Function Documentation

7.65.3.1 virtual void Wombat::MamaSymbolSourceCallback::onSymbol (MamaSymbolSource * *symbolSource*, const char * *symbol*, void * *closure*) [pure virtual]

Method invoked when a symbol arrives at a symbol source.

Parameters:

symbolSource The symbolSource.

symbol The newly sourced symbol

closure The closure associated with the symbol source.

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

- MamaSymbolSourceCallback.h

7.66 Wombat::MamaSymbolStoreSaveCallback Class Reference

The MamaSymbolStoreSaveCallback interface.

`#include <MamaSymbolStoreSaveCallback.h>`

Public Member Functions

- virtual ~MamaSymbolStoreSaveCallback ()
- virtual const char * savefilterConverter (const char *symbol)=0

  Method invoked when a symbol is being saved to file.

7.66.1 Detailed Description

The MamaSymbolStoreSaveCallback interface.

An Object implementing this interface can be passed to the saveSymbolsToFile() function as a mechanism for filtering / transforming symbols being saved to files.

7.66.2 Constructor & Destructor Documentation

7.66.2.1 virtual Wombat::MamaSymbolStoreSaveCallback::~MamaSymbolStoreSaveCallback () [virtual]

```
38 {};
```

7.66.3 Member Function Documentation

7.66.3.1 virtual const char* Wombat::MamaSymbolStoreSaveCallback::savefilterConverter (const char * *symbol*) [pure virtual]

Method invoked when a symbol is being saved to file.

Parameters:

symbol The symbol just added to the store.

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

- MamaSymbolStoreSaveCallback.h

7.67 Wombat::MamaTimer Class Reference

A repeating timer.

`#include <MamaTimer.h>`

Public Member Functions

- MamaTimer (void)
- virtual ~MamaTimer (void)
- virtual void create (MamaQueue *queue, MamaTimerCallback *callback, mama_f64_t interval, void *closure=NULL)

  Create a repeating timer.

- virtual void destroy ()

  Destroy (stop) the timer.

- virtual bool isActive () const

  Return whether the timer is active.

- virtual void reset ()

  Reset the timer to the beginning of the interval.

- virtual void setInterval (mama_f64_t intervalSeconds)

  Set the timer to use a different interval (and reset to the beginning of that interval).

- virtual mama_f64_t getInterval () const

  Get the current timer interval.

- virtual MamaTimerCallback * getCallback () const

  Return the callback for the timer.

- virtual void * getClosure () const

  Return the closure for the timer.

- mamaTimer getCValue ()
- const mamaTimer getCValue () const

7.67.1 Detailed Description

A repeating timer.

The callback will be repeatedly called at the specified interval until the timer is destroyed. To restart the timer after destroying it if destroy(), use create(). To reset the timer to the beginning of the given interval, use reset(). To set the timer to a different interval, use setInterval().

The MAMA timer relies on underlying middleware so its resolution is also dependent on the middleware.

7.67.2 Constructor & Destructor Documentation

7.67.2.1 Wombat::MamaTimer::MamaTimer (void)

7.67.2.2 virtual Wombat::MamaTimer::~MamaTimer (void) [virtual]

7.67.3 Member Function Documentation

7.67.3.1 virtual void Wombat::MamaTimer::create (MamaQueue * *queue*, MamaTimerCallback * *callback*, mama_f64_t *interval*, void * *closure* = NULL) [virtual]

Create a repeating timer.

The interval is in seconds.

The queue is the queue from which the timer event will be dispatched.

Parameters:

queue The queue.

callback The callback.

interval The interval in seconds.

closure The caller supplied closure.

7.67.3.2 virtual void Wombat::MamaTimer::destroy () [virtual]

Destroy (stop) the timer.

This function must be called from the same thread dispatching on the associated event queue unless both the default queue and dispatch queue are not actively dispatching.

7.67.3.3 virtual bool Wombat::MamaTimer::isActive () const [virtual]

Return whether the timer is active.

7.67.3.4 virtual void Wombat::MamaTimer::reset () [virtual]

Reset the timer to the beginning of the interval.

7.67.3.5 virtual void Wombat::MamaTimer::setInterval (mama_f64_t *intervalSeconds*) [virtual]

Set the timer to use a different interval (and reset to the beginning of that interval).

7.67.3.6 virtual mama_f64_t Wombat::MamaTimer::getInterval () const [virtual]

Get the current timer interval.

7.67.3.7 virtual MamaTimerCallback* Wombat::MamaTimer::getCallback () const [virtual]

Return the callback for the timer.

Returns:

the callback.

7.67.3.8 virtual void* Wombat::MamaTimer::getClosure () const [virtual]

Return the closure for the timer.

Returns:

the closure.

7.67.3.9 mamaTimer Wombat::MamaTimer::getCValue ()

7.67.3.10 const mamaTimer Wombat::MamaTimer::getCValue () const

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

- MamaTimer.h

7.68 Wombat::MamaTimerCallback Class Reference

Subclass this to receive timer notifications.

`#include <MamaTimerCallback.h>`

Public Member Functions

- virtual ~MamaTimerCallback (void)
- virtual void onTimer (MamaTimer *timer)=0
- virtual void onDestroy (MamaTimer *timer, void *closure)

7.68.1 Detailed Description

Subclass this to receive timer notifications.

7.68.2 Constructor & Destructor Documentation

7.68.2.1 virtual Wombat::MamaTimerCallback::~MamaTimerCallback (void) [virtual]

```
37 {}
```

7.68.3 Member Function Documentation

7.68.3.1 virtual void Wombat::MamaTimerCallback::onTimer (MamaTimer * *timer*) [pure virtual]

7.68.3.2 virtual void Wombat::MamaTimerCallback::onDestroy (MamaTimer * *timer*, void * *closure*) [virtual]

```
39 {};
```

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

- MamaTimerCallback.h

7.69 Wombat::MamaTimeZone Class Reference

A time zone representation to make conversion of timestamps to and from particular time zones more convenient.

`#include <MamaTimeZone.h>`

Public Member Functions

- MamaTimeZone ()

  Constructor.

- MamaTimeZone (const char *tz)

  Constructor.

- MamaTimeZone (const MamaTimeZone &copy)

  Copy constructor.

- ~MamaTimeZone ()

  Destructor.

- MamaTimeZone & operator= (const MamaTimeZone &rhs)

  Assignment operator.

- void set (const char *tz)

  Assign new timezones to this object.

- void clear ()

  Clear this object.

- const char * tz () const

  Return the time zone string.

- mama_i32_t offset () const

  Return the offset from UTC (in seconds).

- void check ()

  Check (recalculate) the UTC offset in case it has changed due to daylight savings adjustments.

- mamaTimeZone getCValue ()
- const mamaTimeZone getCValue () const

Static Public Member Functions

- static const MamaTimeZone & local ()

  Return a reference to a MamaTimeZone corresponding to the local time zone.

- static const MamaTimeZone & utc ()

  Return a reference to a MamaTimeZone corresponding to UTC time zone.

- static const MamaTimeZone & usEastern ()

  Return a reference to a MamaTimeZone corresponding to the US Eastern time zone.

- static void setScanningInterval (mama_f64_t seconds)

  Use to set the interval of the thread updating each MamaTimeZone instance offset.

7.69.1 Detailed Description

A time zone representation to make conversion of timestamps to and from particular time zones more convenient.

Note: The addition of instance monitoring to the MamaTimeZone implementation has resulted in the following limitation in its usage. Do not create short lived objects of this type on the method stack or delete long lived objects before program termination. Pointers to all instances are maintained in a global vector. At the moment there is no mechanism by which we can detect deleted objects or those which are popped off the method stack. An internal thread will always iterate over all objects ever created. A call to an object removed from the stack will result in nondeterminable behaviour. Pointers could be stored in a map against a unique object id; however, addition and removal from the map would have to be synchronized which would impact on performance.

7.69.2 Constructor & Destructor Documentation

7.69.2.1 Wombat::MamaTimeZone::MamaTimeZone ()

Constructor.

7.69.2.2 Wombat::MamaTimeZone::MamaTimeZone (const char * *tz*)

Constructor.

NULL argument is equivalent to local timezone.

7.69.2.3 Wombat::MamaTimeZone::MamaTimeZone (const MamaTimeZone & *copy*)

Copy constructor.

7.69.2.4 Wombat::MamaTimeZone::~MamaTimeZone ()

Destructor.

7.69.3 Member Function Documentation

7.69.3.1 static const MamaTimeZone& Wombat::MamaTimeZone::local () [static]

Return a reference to a MamaTimeZone corresponding to the local time zone.

7.69.3.2 static const MamaTimeZone& Wombat::MamaTimeZone::utc () [static]

Return a reference to a MamaTimeZone corresponding to UTC time zone.

7.69.3.3 static const MamaTimeZone& Wombat::MamaTimeZone::usEastern () [static]

Return a reference to a MamaTimeZone corresponding to the US Eastern time zone.

7.69.3.4 MamaTimeZone& Wombat::MamaTimeZone::operator= (const MamaTimeZone & *rhs*)

Assignment operator.

7.69.3.5 void Wombat::MamaTimeZone::set (const char * *tz*)

Assign new timezones to this object.

7.69.3.6 void Wombat::MamaTimeZone::clear ()

Clear this object.

7.69.3.7 const char* Wombat::MamaTimeZone::tz () const

Return the time zone string.

7.69.3.8 mama_i32_t Wombat::MamaTimeZone::offset () const

Return the offset from UTC (in seconds).

Can be positive or negative, depending upon the direction.

7.69.3.9 void Wombat::MamaTimeZone::check ()

Check (recalculate) the UTC offset in case it has changed due to daylight savings adjustments.

7.69.3.10 mamaTimeZone Wombat::MamaTimeZone::getCValue ()

```
96 { return myCimpl; }
```

7.69.3.11 const mamaTimeZone Wombat::MamaTimeZone::getCValue () const

```
97 { return myCimpl; }
```

7.69.3.12 static void Wombat::MamaTimeZone::setScanningInterval (mama_f64_t *seconds*) [static]

Use to set the interval of the thread updating each MamaTimeZone instance offset.

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

- MamaTimeZone.h

7.70 Wombat::MamaTransport Class Reference

The MamaTransport class provides market data functionality.

`#include <MamaTransport.h>`

Public Member Functions

- MamaTransport ()

  Construct a MamaTransport.

- virtual ~MamaTransport ()
- MamaTransport (mamaTransport cTransport)

  Construct a MamaTransport that wraps a mamaTransport from the C API.

- void create (const char *name, mamaBridge bridgeImpl)

  Create a transport.

- void setName (const char *name)

  set the transport name.

- const char * getName () const

  get the transport name.

- const char * getMiddleware () const

  get the middleware name.

- double getOutboundThrottle (mamaThrottleInstance instance=MAMA_THROTTLE_DEFAULT) const

  Return the outbound throttle rate in messages/second.

- void setOutboundThrottle (double outboundThrottle, mamaThrottleInstance instance=MAMA_THROTTLE_DEFAULT)

  Set the throttle rate for outbound message.

- void setTransportTopicCallback (MamaTransportTopicEventCallback *callback)

  Set the transport topic callback.

- void setTransportCallback (MamaTransportCallback *callback)

  Set the transport callback.

- MamaTransportCallback * getTransportCallback ()

Get the transport callback.

- void setSymbolMap (const MamaSymbolMap *mapper)

  Set the symbology mapping class.

- const MamaSymbolMap * getSymbolMap () const

  Return the symbology mapping class.

- void setDescription (const char *description)

  Set the description for the transport.

- const char * getDescription () const

  Return the description string for the transport.

- MamaConnection * findConnection (const char *IpAddress, uint16_t port)
- virtual void getAllConnections (MamaConnection **&list, uint32_t &len)
- virtual void freeAllConnections (MamaConnection **list, uint32_t len)
- virtual void getAllServerConnections (MamaServerConnection **&list, uint32_t &len)
- virtual void freeAllServerConnections (MamaServerConnection **list, uint32_t &len)
- void setInvokeQualityForAllSubscs (bool invokeQualityForAllSubscs)

  Set whether to invoke the quality callback for all subscriptions whenever a data quality event occurs (the default), or whether to invoke the quality callback only when data subsequently arrives for a subscription.

- bool getInvokeQualityForAllSubscs () const

  Get whether the transport has been set to invoke the quality callback for all subscriptions whenever a data quality event occurs.

- mamaQuality getQuality () const

  Get the data quality for the transport.

- void requestConflation () const

  Request conflation for the transport.

- void requestEndConflation () const

  Request an end to conflation for the transport.

- mamaTransport getCValue ()
- const mamaTransport getCValue () const
- void * getNativeTransport (int index)

  Return the underlying native transport.

- void disableRefresh (bool disable)

  Disable refreshing of subscriptions on this transport.

Public Attributes

- MamaTransportImpl * mPimpl

7.70.1 Detailed Description

The MamaTransport class provides market data functionality.

7.70.2 Constructor & Destructor Documentation

7.70.2.1 Wombat::MamaTransport::MamaTransport ()

Construct a MamaTransport.

Call create to create the transport.

7.70.2.2 virtual Wombat::MamaTransport::~MamaTransport () [virtual]

7.70.2.3 Wombat::MamaTransport::MamaTransport (mamaTransport *cTransport*)

Construct a MamaTransport that wraps a mamaTransport from the C API.

Mama uses this internally. C++ Applications should create C++ MamaTransport objects through the no-argument constructor, and call MamaTransport::create().

MamaTransport objects created with this method do not deallocate or destroy the underlying c Transport because that responsibility lies with the creator.

7.70.3 Member Function Documentation

7.70.3.1 void Wombat::MamaTransport::create (const char * *name*, mamaBridge *bridgeImpl*)

Create a transport.

Platform specific parameters are read from the properties file. The parameters are dependent on the underlying messaging transport.

Parameters:

name The transport name

bridgeImpl The middleware-specific bridge structure

7.70.3.2 void Wombat::MamaTransport::setName (const char * *name*)

set the transport name.

The name string is copied by the object.

Parameters:

name The transport name.

7.70.3.3 const char* Wombat::MamaTransport::getName () const

get the transport name.

7.70.3.4 const char* Wombat::MamaTransport::getMiddleware () const

get the middleware name.

7.70.3.5 double Wombat::MamaTransport::getOutboundThrottle (mamaThrottleInstance *instance* = MAMA_THROTTLE_DEFAULT) const

Return the outbound throttle rate in messages/second.

Returns:

The throttle rate.

7.70.3.6 void Wombat::MamaTransport::setOutboundThrottle (double *outboundThrottle*, mamaThrottleInstance *instance* = MAMA_THROTTLE_DEFAULT)

Set the throttle rate for outbound message.

This rate controls the rate at which methods sent with sendWithThrottle (void) are sent. A value of 0.0 disables throttling.

Parameters:

outboundThrottle The rate in messages/second.

instance the mamaThrottleInstance to use

7.70.3.7 void Wombat::MamaTransport::setTransportTopicCallback (MamaTransportTopicEventCallback * *callback*)

Set the transport topic callback.

7.70.3.8 void Wombat::MamaTransport::setTransportCallback (MamaTransportCallback * *callback*)

Set the transport callback.

7.70.3.9 MamaTransportCallback* Wombat::MamaTransport::getTransportCallback ()

Get the transport callback.

7.70.3.10 void Wombat::MamaTransport::setSymbolMap (const MamaSymbolMap * *mapper*)

Set the symbology mapping class.

Parameters:

mapper A symbol mapping class.

7.70.3.11 const MamaSymbolMap* Wombat::MamaTransport::getSymbolMap () const

Return the symbology mapping class.

7.70.3.12 void Wombat::MamaTransport::setDescription (const char * *description*)

Set the description for the transport.

The description string is copied by the object.

Parameters:

description The transport description.

7.70.3.13 const char* Wombat::MamaTransport::getDescription () const

Return the description string for the transport.

Do not alter or free the string returned by this method.

Returns:

const char* The transport description.

7.70.3.14 MamaConnection* Wombat::MamaTransport::findConnection (const char * *IpAddress*, uint16_t *port*)

7.70.3.15 virtual void Wombat::MamaTransport::getAllConnections (MamaConnection **& *list*, uint32_t & *len*) [virtual]

7.70.3.16 virtual void Wombat::MamaTransport::freeAllConnections (MamaConnection ** *list*, uint32_t *len*) [virtual]

7.70.3.17 virtual void Wombat::MamaTransport::getAllServerConnections (MamaServerConnection **& *list*, uint32_t & *len*) [virtual]

7.70.3.18 virtual void Wombat::MamaTransport::freeAllServerConnections (MamaServerConnection ** *list*, uint32_t & *len*) [virtual]

7.70.3.19 void Wombat::MamaTransport::setInvokeQualityForAllSubscs (bool *invokeQualityForAllSubscs*)

Set whether to invoke the quality callback for all subscriptions whenever a data quality event occurs (the default), or whether to invoke the quality callback only when data subsequently arrives for a subscription.

If set to true, an onQuality callback will be invoked for a subscription whenever a data quality event occurs on the transport, even in between updates for that description. If set to false, the onQuality callback will not be called when the data quality event occurs on the transport. However, it will still be invoked when an update arrives for the subscription.

Parameters:

invokeQualityForAllSubscs Whether to invoke quality callback for all subscriptions

7.70.3.20 bool Wombat::MamaTransport::getInvokeQualityForAllSubscs () const

Get whether the transport has been set to invoke the quality callback for all subscriptions whenever a data quality event occurs.

Returns:

Whether transport has been set to invoke quality callback for all subscriptions

7.70.3.21 mamaQuality Wombat::MamaTransport::getQuality () const

Get the data quality for the transport.

Currently only supported for the Tibco RV middleware. Returns OK for all other middlewares.

Returns:

The Quality of the transport

7.70.3.22 void Wombat::MamaTransport::requestConflation () const

Request conflation for the transport.

Currently only supported for WMW.

7.70.3.23 void Wombat::MamaTransport::requestEndConflation () const

Request an end to conflation for the transport.

Currently only supported for WMW.

7.70.3.24 mamaTransport Wombat::MamaTransport::getCValue ()

7.70.3.25 const mamaTransport Wombat::MamaTransport::getCValue () const

7.70.3.26 void* Wombat::MamaTransport::getNativeTransport (int *index*)

Return the underlying native transport.

Applications should avoid this method if possible as it may result in non-portable, middleware specific code. Callers must cast the nativeTport to the appropriate type.

Note: this method returns the underlying C construct not a C++ object.

7.70.3.27 void Wombat::MamaTransport::disableRefresh (bool *disable*)

Disable refreshing of subscriptions on this transport.

7.70.4 Member Data Documentation

7.70.4.1 MamaTransportImpl* Wombat::MamaTransport::mPimpl

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

- MamaTransport.h

7.71 Wombat::MamaTransportCallback Class Reference

Transport callback.

`#include <MamaTransport.h>`

Public Member Functions

- virtual ~MamaTransportCallback ()
- virtual void onDisconnect (MamaTransport *transport, const void *platformInfo)

  Invoked on a publisher when a subscriber disconnects.

- virtual void onReconnect (MamaTransport *transport, const void *platformInfo)

  Invoked when the transport reconnects.

- virtual void onQuality (MamaTransport *transport, short cause, const void *platformInfo)=0

  Invoked when the quality of this transport changes.

- virtual void onConnect (MamaTransport *transport, const void *platformInfo)

  Invoked on the subscriber when the transport connects.

- virtual void onAccept (MamaTransport *transport, const void *platformInfo)

  Invoked on the publisher when the transport accepts a connection.

- virtual void onAcceptReconnect (MamaTransport *transport, const void *platformInfo)

  Invoked on the publisher when the transport accepts a reconnection.

- virtual void onPublisherDisconnect (MamaTransport *transport, const void *platformInfo)

  Invoked on the subscriber when the transport disconnects from the publisher.

- virtual void onNamingServiceConnect (MamaTransport *transport, const void *platformInfo)

  Invoked on the subscriber when the naming service connects.

- virtual void onNamingServiceDisconnect (MamaTransport *transport, const void *platformInfo)

  Invoked on the subscriber when the naming service disconnects.

7.71.1 Detailed Description

Transport callback.

7.71.2 Constructor & Destructor Documentation

7.71.2.1 virtual Wombat::MamaTransportCallback::~MamaTransportCallback () [virtual]

```
72          {
73          };
```

7.71.3 Member Function Documentation

7.71.3.1 virtual void Wombat::MamaTransportCallback::onDisconnect (MamaTransport * *transport*, const void * *platformInfo*) [virtual]

Invoked on a publisher when a subscriber disconnects.

Parameters:

transport The transport which has disconnected.

platformInfo Info associated with the event.

The cause and platformInfo are supplied only by some middlewares. The information provided by platformInfo is middleware specific. The following middlewares are supported:

tibrv: provides the char* version of the tibrv advisory message. wmw: provides a pointer to a C mamaConnection struct for the event

```
91          {
92              return;
93          }
```

7.71.3.2 virtual void Wombat::MamaTransportCallback::onReconnect (MamaTransport * *transport*, const void * *platformInfo*) [virtual]

Invoked when the transport reconnects.

Parameters:

transport The transport which has reconnected.

platformInfo Info associated with the event.

The cause and platformInfo are supplied only by some middlewares. The information provided by platformInfo is middleware specific. The following middlewares are supported:

tibrv: provides the char* version of the tibrv advisory message. wmw: provides a pointer to a C mamaConnection struct for the event

```
111         {
112             return;
113         }
```

7.71.3.3 virtual void Wombat::MamaTransportCallback::onQuality (MamaTransport * *transport*, short *cause*, const void * *platformInfo*) [pure virtual]

Invoked when the quality of this transport changes.

Parameters:

transport The transport on which the quality has changed.

cause The cause of the quality event.

platformInfo Info associated with the quality event.

The cause and platformInfo are supplied only by some middlewares. The information provided by platformInfo is middleware specific. The following middlewares are supported:

tibrv: provides the char* version of the tibrv advisory message.

7.71.3.4 virtual void Wombat::MamaTransportCallback::onConnect (MamaTransport * *transport*, const void * *platformInfo*) [virtual]

Invoked on the subscriber when the transport connects.

Parameters:

transport The transport which has connected.

platformInfo Info associated with the event.

The cause and platformInfo are supplied only by some middlewares. The information provided by platformInfo is middleware specific. The following middlewares are supported:

wmw: provides a pointer to a C mamaConnection struct for the event

```
148         {
149             return;
150         }
```

7.71.3.5 virtual void Wombat::MamaTransportCallback::onAccept (MamaTransport * *transport*, const void * *platformInfo*) [virtual]

Invoked on the publisher when the transport accepts a connection.

Parameters:

transport The transport which has accepted.

platformInfo Info associated with the event.

The cause and platformInfo are supplied only by some middlewares. The information provided by platformInfo is middleware specific. The following middlewares are supported:

wmw: provides a pointer to a C mamaConnection struct for the event

```
167         {
168             return;
169         }
```

7.71.3.6 virtual void Wombat::MamaTransportCallback::onAcceptReconnect (MamaTransport * *transport*, const void * *platformInfo*) [virtual]

Invoked on the publisher when the transport accepts a reconnection.

Parameters:

transport The transport which has reconnected on

platformInfo Info associated with the event.

The cause and platformInfo are supplied only by some middlewares. The information provided by platformInfo is middleware specific. The following middlewares are supported:

wmw: provides a pointer to a C mamaConnection struct for the event

```
186         {
187             return;
188         }
```

7.71.3.7 virtual void Wombat::MamaTransportCallback::onPublisher-Disconnect (MamaTransport * *transport*, const void * *platformInfo*) [virtual]

Invoked on the subscriber when the transport disconnects from the publisher.

Parameters:

transport The transport which has disconnected on

platformInfo Info associated with the event.

The cause and platformInfo are supplied only by some middlewares. The information provided by platformInfo is middleware specific. The following middlewares are supported:

wmw: provides a pointer to a C mamaConnection struct for the event

```
205         {
206             return;
207         }
```

7.71.3.8 virtual void Wombat::MamaTransportCallback::onNamingService-Connect (MamaTransport * *transport*, const void * *platformInfo*) [virtual]

Invoked on the subscriber when the naming service connects.

Parameters:

transport The transport which has connected.

platformInfo Info associated with the event.

```
218         {
219             return;
220         }
```

7.71.3.9 virtual void Wombat::MamaTransportCallback::onNamingService-Disconnect (MamaTransport * *transport*, const void * *platformInfo*) [virtual]

Invoked on the subscriber when the naming service disconnects.

Parameters:

transport The transport which has connected.

platformInfo Info associated with the event.

```
231         {
232             return;
233         }
```

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

- MamaTransport.h

7.72 Wombat::MamaTransportMap Class Reference

`#include <MamaTransportMap.h>`

Static Public Member Functions

- static MamaTransport * findOrCreate (const char *transportName, const mamaBridge bridge)

  Find the transport by name.

- static MamaTransport * find (const char *transportName, const mamaBridge bridge)

  Find the transport by name.

7.72.1 Member Function Documentation

7.72.1.1 static MamaTransport* Wombat::MamaTransportMap::findOrCreate (const char * *transportName*, const mamaBridge *bridge*) [static]

Find the transport by name.

If no transport by the given name has been requested before, a new instance of a MamaTransport is created using the bridge specified. This method will create exactly one MamaTransport object instance for each name passed.

7.72.1.2 static MamaTransport* Wombat::MamaTransportMap::find (const char * *transportName*, const mamaBridge *bridge*) [static]

Find the transport by name.

If no transport by the given name exists, the default transport is returned.

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

- MamaTransportMap.h

7.73 Wombat::MamaTransportTopicEventCallback Class Reference

TransportTopicEvent callback.

`#include <MamaTransport.h>`

Public Member Functions

- virtual ~MamaTransportTopicEventCallback ()
- virtual void onTopicSubscribe (MamaTransport *transport, const char *topic, const void *platformInfo)

  Invoked when a topic is subscribed to.

- virtual void onTopicUnsubscribe (MamaTransport *transport, const char *topic, const void *platformInfo)

7.73.1 Detailed Description

TransportTopicEvent callback.

7.73.2 Constructor & Destructor Documentation

7.73.2.1 virtual Wombat::MamaTransportTopicEventCallback::~MamaTransportTopicEventCallback () [virtual]

```
43          {
44          };
```

7.73.3 Member Function Documentation

7.73.3.1 virtual void Wombat::MamaTransportTopicEventCallback::onTopicSubscribe (MamaTransport * *transport*, const char * *topic*, const void * *platformInfo*) [virtual]

Invoked when a topic is subscribed to.

```
52          {
53              return;
54          }
```

7.73.3.2 virtual void Wombat::MamaTransportTopicEventCallback::onTopicUnsubscribe (MamaTransport * *transport*, const char * *topic*, const void * *platformInfo*) [virtual]

```
59          {
60              return;
61          }
```

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

- MamaTransport.h

Chapter 8

MAMA C++ API File Documentation

8.1 MamaBasicSubscription.h File Reference

```
#include "mama/mama.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaBasicSubscription

  The `MamaBasicSubscription` *interface represents a subscription to a topic with no market data semantics.*

8.2 MamaBasicSubscriptionCallback.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaBasicSubscriptionCallback

  The message callback interface for basic subscriptions.

8.3 MamaBasicWildCardSubscription.h File Reference

```
#include "mama/mama.h"
#include "mama/MamaBasicSubscription.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaBasicWildCardSubscription

  The `MamaBasicWildCardSubscription` *interface represents a subscription to a topic with no market data semantics.*

8.4 MamaBasicWildCardSubscriptionCallback.h File Reference

```
#include "mama/mamacpp.h"
#include "mama/MamaBasicWildCardSubscription.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaBasicWildCardSubscriptionCallback

  The message callback interface for basic subscriptions.

8.5 MamaBridgeCallback.h File Reference

```
#include "mama/mama.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaBridgeCallback
  Bridge callback.

Defines

- #define MAMA_BRIDEG_CALLBACK_CPP_H

8.5.1 Define Documentation

8.5.1.1 #define MAMA_BRIDEG_CALLBACK_CPP_H

8.6 mamacpp.h File Reference

```
#include <stdio.h>
#include <cstring>
#include <mama/mama.h>
#include <mama/MamaBridgeCallback.h>
#include <mama/MamaMsg.h>
#include <mama/MamaQueueGroup.h>
#include <mama/MamaBasicSubscription.h>
#include <mama/MamaBasicSubscriptionCallback.h>
#include <mama/MamaBasicWildCardSubscription.h>
#include <mama/MamaBasicWildCardSubscriptionCallback.h>
#include <mama/MamaSubscription.h>
#include <mama/MamaSubscriptionCallback.h>
#include <mama/MamaTransport.h>
#include <mama/MamaPublisher.h>
#include <mama/MamaInboxCallback.h>
#include <mama/MamaInbox.h>
#include <mama/MamaQueue.h>
#include <mama/MamaQueueEnqueueCallback.h>
#include <mama/MamaQueueMonitorCallback.h>
#include <mama/MamaQueueEventCallback.h>
#include <mama/MamaDispatcher.h>
#include <mama/MamaTimerCallback.h>
#include <mama/MamaTimer.h>
#include <mama/MamaIoCallback.h>
#include <mama/MamaIo.h>
#include <mama/MamaDictionaryCallback.h>
#include <mama/MamaDictionary.h>
#include <mama/MamaFieldDescriptor.h>
#include <mama/MamaDateTime.h>
```

```
#include <mama/MamaPrice.h>
#include <mama/MamaMsgFieldIterator.h>
#include <mama/MamaMsgField.h>
#include <mama/MamaStatus.h>
#include <mama/MamaSymbolMap.h>
#include <mama/MamaSymbolMapFile.h>
#include <mama/MamaLogFile.h>
#include <mama/MamaSymbolSourceCallback.h>
#include <mama/MamaSymbolSource.h>
#include <mama/MamaMsgQual.h>
#include <mama/msgstatus.h>
#include <mama/types.h>
#include <mama/MamaSendCompleteCallback.h>
#include <mama/MamaSource.h>
#include <mama/MamaSourceManager.h>
#include <mama/MamaSourceGroup.h>
#include <mama/MamaSourceGroupManager.h>
#include <mama/MamaStatsCollector.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaLogFileCallback

  Subclass this to receive log notifications.

- class Wombat::MamaStartCallback

  Callback object passed to Mama::startBackground().

- class Wombat::Mama

  The `Mama` *class provides methods global initialization and manipulating global options.*

8.7 MamaDateTime.h File Reference

```
#include <mama/datetime.h>
#include <mama/MamaTimeZone.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaDateTime

  A date/time representation with additional hints for precision, advanced output formatting and support for time zone conversion (using the MamaTimeZone type).

Functions

- mama_f64_t diffSeconds (const Wombat::MamaDateTime &t1, const Wombat::MamaDateTime &t0)
- mama_f64_t diffSecondsSameDay (const Wombat::MamaDateTime &t1, const Wombat::MamaDateTime &t0)
- mama_i64_t diffMicroseconds (const Wombat::MamaDateTime &t1, const Wombat::MamaDateTime &t0)

8.7.1 Function Documentation

8.7.1.1 mama_f64_t diffSeconds (const Wombat::MamaDateTime & *t1*, const Wombat::MamaDateTime & *t0*)

8.7.1.2 mama_f64_t diffSecondsSameDay (const Wombat::MamaDateTime & *t1*, const Wombat::MamaDateTime & *t0*)

8.7.1.3 mama_i64_t diffMicroseconds (const Wombat::MamaDateTime & *t1*, const Wombat::MamaDateTime & *t0*)

8.8 MamaDictionary.h File Reference

```
#include <mama/mamacpp.h>
#include <mama/MamaFieldDescriptor.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaDictionary

  The `MamaDictionary` *class maps field identifiers (FIDs) to human readable strings.*

8.9 MamaDictionaryCallback.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaDictionaryCallback

  The `WombatDictionaryCallback` *receives notification regarding the population of the data dictionary.*

8.10 MamaDispatcher.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaDispatcher

  The dispatcher dispatches events from a queue until it is destroyed or MamaQueue->stopDispatch () is called.

8.11 MamaDQPublisher.h File Reference

```
#include "mama/msgstatus.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaDQPublisher

8.12 MamaDQPublisherManager.h File Reference

```
#include "mama/MamaDQPublisherManagerCallback.h"
#include "mama/dqpublishermanager.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaPublishTopic
- class Wombat::MamaDQPublisherManager

8.13 MamaDQPublisherManagerCallback.h File Reference

```
#include "mama/mamacpp.h"
#include "mama/msgtype.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaDQPublisherManagerCallback

8.14 MamaFieldDescriptor.h File Reference

```
#include <mama/mamacpp.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaFieldDescriptor

  The `MamaFieldDescriptor` *class describes a field within a* `MamaDictionary`.

8.15 MamaFt.h File Reference

```
#include <mama/ft.h>
#include <mama/MamaQueue.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaFtMemberCallback
- class Wombat::MamaFtMember
- class Wombat::MamaMulticastFtMember
- class Wombat::MamaBridgeFtMember

8.16 MamaInbox.h File Reference

```
#include <mama/mamacpp.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaInbox

  Used for sending messages requesting a direct reply.

8.17 MamaInboxCallback.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaInboxCallback

  The MamaInboxCallback gets invoked when a message arrives in an inbox or when inbox related errors arise.

8.18 MamaIo.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaIo

  A repeating IO.

8.19 MamaIoCallback.h File Reference

```
#include "mama/mama.h"
#include <mama/io.h>
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaIoCallback

  Subclass this to receive IO notifications.

8.20 MamaLogFile.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaLogFile

  The `MamaLogFile` *class provides a single interface for the configuration and control of* `Mama` *logging activity.*

8.21 MamaMsg.h File Reference

```
#include "mama/mamacpp.h"
#include "mama/MamaMsgField.h"
#include "mama/types.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaMsgIterator
  Iterator class for mamaMsg.
- class Wombat::MamaMsg
  MAMA message representation.

8.22 MamaMsgField.h File Reference

```
#include <mama/mamacpp.h>
#include <vector>
#include <functional>
```

Namespaces

- namespace Wombat
- namespace std

Classes

- class Wombat::MamaMsgField

  MamaMsg field representation.

8.23 MamaMsgFieldIterator.h File Reference

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaMsgFieldIterator

  Callback class for iterating over all fields in a message.

8.24 MamaMsgQual.h File Reference

```
#include <mama/msgqualifier.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaMsgQual

  The MamaMsgQual class is a wrapper/utility class which provides useful interrogation, comparison and manipulation facilities for the Mama Message Qualifier data field (wMsgQual) which is a "bit-map" used to convey duplicate, delayed and out-of-sequence information about messages.

8.25 MamaPrice.h File Reference

```
#include <mama/price.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaPrice

  MamaPrice is a special data type for representing floating point numbers that often require special formatting for display purposes, such as prices.

8.26 MamaPublisher.h File Reference

```
#include "mama/mamacpp.h"
#include "mama/MamaSendCompleteCallback.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaPublisher

  The publisher class publishes messages to basic or market data subscriptions depending on the underlying transport.

8.27 MamaQueue.h File Reference

```
#include <mama/mamacpp.h>
#include <mama/queue.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaQueue

  Queue allows applications to dispatch events in order with multiple threads using a single MamaDispatcher for each queue.

8.28 MamaQueueEnqueueCallback.h File Reference

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaQueueEnqueueCallback

  Callback interface for the MamaQueue::setEnqueueCallback () method.

8.29 MamaQueueEventCallback.h File Reference

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaQueueEventCallback

  Definition of the callback method for when a user added event fires.

8.30 MamaQueueGroup.h File Reference

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaQueueGroup

  A simple class for allocating subscriptions amongst multiple queues in a round robin.

8.31 MamaQueueMonitorCallback.h File Reference

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaQueueMonitorCallback

  Receive callbacks when certain conditions for the MamaQueue are met.

8.32 MamaReservedFields.h File Reference

```
#include <mama/mamacpp.h>
#include <mama/reservedfields.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaReservedFields

8.33 MamaSendCompleteCallback.h File Reference

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSendCompleteCallback

  Callback interface for use with the MamaPublisher.sendWithThrottle() and MamaPublisher.sendFromInboxWithThrottle() methods.

8.34 MamaSource.h File Reference

```
#include <mama/source.h>
#include <mama/MamaSourceManager.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSubscriptionIteratorCallback
- class Wombat::MamaSource

  A MAMA source maintains information about a data source, including the quality of the data coming from that source.

8.35 MamaSourceDerivative.h File Reference

```
#include <mama/MamaSource.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSourceDerivative

  A MamaSourceDerivative provides a reference to a common MamaSource object but can have attributes (such as the quality state) overridden.

8.36 MamaSourceGroup.h File Reference

```
#include <mama/types.h>
#include <mama/MamaSourceStateChangeCallback.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSourceGroup

  A MAMA source group monitors a set of MAMA sources that presumably provide a duplicate set of data.

- class Wombat::MamaSourceGroup::iterator

8.37 MamaSourceGroupManager.h File Reference

```
#include <mama/types.h>
#include <mama/config.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSourceGroupManager

  A MAMA source group manager monitors a set of MAMA source groups.

- class Wombat::MamaSourceGroupManager::iterator

8.38 MamaSourceManager.h File Reference

```
#include <mama/sourceman.h>
#include <mama/types.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSourceManager

  A MAMA source manager maintains information about a set of data sources, including the quality of the data coming from those sources.

- class Wombat::MamaSourceManager::iterator

8.39 MamaSourceStateChangeCallback.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSourceStateChangeCallback

  Applications can register with MamaSourceGroup to receive state change notifications when the state of sources within the group has changed.

8.40 MamaStat.h File Reference

```
#include "mama/types.h"
#include "mama/mamacpp.h"
#include "mama/stat.h"
#include "mama/statfields.h"
#include "mama/statscollector.h"
#include "mama/MamaStatsCollector.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaStat

8.41 MamaStatsCollector.h File Reference

```
#include "mama/types.h"
#include "mama/statscollector.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaStatsCollector

8.42 MamaStatus.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaStatus

8.43 MamaSubscription.h File Reference

```
#include "mama/MamaBasicSubscription.h"
#include "mama/subscription.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSubscription

  The `MamaSubscription` *interface represents a subscription to a topic.*

8.44 MamaSubscriptionCallback.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSubscriptionCallback

  The message callback interface.

8.45 MamaSymbolList.h File Reference

```
#include <mama/mamacpp.h>
#include <mama/MamaSymbolListMember.h>
#include <mama/MamaSymbolListHandlerTypes.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSymbolList

  MamaSymbolList manages a list of MAMA symbols and related attributes.

8.46 MamaSymbolListFile.h File Reference

```
#include <mama/MamaSymbolList.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSymbolListFile

  MamaSymbolListFile is a file based symbol list with the ability to detect external changes to the file.

8.47 MamaSymbolListHandlerTypes.h File Reference

```
#include <mama/mamacpp.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSymbolListIteratorHandler
  The MamaSymbolListIteratorHandler interface.
- class Wombat::MamaSymbolListMembershipHandler
  The MamaSymbolListMembershipHandler interface.
- class Wombat::MamaSymbolListResortHandler
  The MamaSymbolListResortHandler interface.

8.48 MamaSymbolListMember.h File Reference

```
#include <mama/mamacpp.h>
#include <mama/symbollisttypes.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSymbolListMember

  The MamaSymbolListMember class represents the information about the symbol list member.

8.49 MamaSymbolMap.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSymbolMap

  The MamaSymbolMap class provides the ability for MAMA to do client side symbology mapping.

8.50 MamaSymbolMapFile.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSymbolMapFile

  MamaSymbolMapFile is a concrete implementation of a symbology map.

8.51 MamaSymbolSource.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSymbolSource

  MamaSymbolSource defines an interface which all SymbolSources should implement in order to provide a mechanism by which objects implementing the "MamaSymbolSourceCallback::onSymbol" can be registered with the source such that they can be notified of new symbols as they arrive.

8.52 MamaSymbolSourceCallback.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSymbolSourceCallback

  The MamaSymbolSourceCallback interface.

8.53 MamaSymbolStoreSaveCallback.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaSymbolStoreSaveCallback

  The MamaSymbolStoreSaveCallback interface.

8.54 MamaTimer.h File Reference

```
#include <mama/mamacpp.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaTimer

  A repeating timer.

8.55 MamaTimerCallback.h File Reference

```
#include "mama/mamacpp.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaTimerCallback

  Subclass this to receive timer notifications.

8.56 MamaTimeZone.h File Reference

```
#include <mama/timezone.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaTimeZone

  A time zone representation to make conversion of timestamps to and from particular time zones more convenient.

8.57 MamaTransport.h File Reference

```
#include "mama/mama.h"
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaTransportTopicEventCallback

  TransportTopicEvent callback.

- class Wombat::MamaTransportCallback

  Transport callback.

- class Wombat::MamaTransport

  The MamaTransport class provides market data functionality.

8.58 MamaTransportMap.h File Reference

```
#include <mama/config.h>
#include <mama/mama.h>
```

Namespaces

- namespace Wombat

Classes

- class Wombat::MamaTransportMap