MigratoryData Client API for DotNet

Developer's Guide and Reference Manual

November 5, 2019

Contents

Chapter 1

Developer's Guide

This guide includes the following sections:

1.1 Overview

This application programming interface (API) contains all the necessary operations for connecting to a cluster of one or more MigratoryData servers, subscribing to subjects, getting real-time messages for the subscribed subjects, and publishing real-time messages.

Before reading this manual, it is recommended to read *MigratoryData Architecture Guide* (PDF, HTML).

1.2 Creating .NET clients

A typical API usage is as follows:

1.2.1 Step 1 - Include the library.

Import in your application the classes of this API as follows:

```
using com.migratorydata.client;
```

Also, add this API library `migratorydata-client-dotnet.dll`, located in the folder `lib` of this API package, to the references of your .NET application.

1.2.2 Step 2 - Define the log listener for processing the log messages.

The log listener should implement the MigratoryDataLogListener interface.

Use the API call MigratoryDataClient.SetLogListener() to assign an instance of the log listener for getting the logs of the API.

1.2.3 Step 3 - Define the listener for processing the real-time messages and status notifications.

The listener should implement the MigratoryDataListener interface.

Use the API call MigratoryDataClient.SetListener() to assign an instance of the listener for getting inbound messages and status notifications.

1.2.4 Step 4 - Specify the list of MigratoryData servers where to connect to

Use the API method MigratoryDataClient.SetServers() to specify a list of one or more MigratoryData servers to which the client will connect to. In fact, the client will connect to only one of the MigratoryData servers in this list. But, defining two or more MigratoryData servers is recommended to achieve load balancing and failover. Supposing the MigratoryData server – to which the client connected – goes down, then the API will automatically reconnect to another MigratoryData server in the list.

1.2.5 Step 5 Subscribe to subjects and publish messages

Use the API method MigratoryDataClient.Subscribe() to subscribe to subjects and use the API method MigratoryDataClient.Publish() to publish messages.

1.2.6 Step 6 - Handle the real-time messages and status notifications.

Handle the messages received for the subscribed subjects as well as the status notifications in the class you defined at Step 3 above.

1.3 Examples

Examples built with this API are available in the folder `examples` of this API package; start with the README file which explains how to compile and run them.

Chapter 2

Deprecated List

Member MigratoryDataClient.NOTIFY_PUBLISH_NO_SUBSCRIBER

no more is use.

Member MigratoryDataClient.SetServersDownBeforeNotify (int n)

use NotifyAfterReconnectRetries

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

Class Documentation

4.1 MigratoryDataClient Class Reference

This class implements all the necessary operations for connecting to a cluster of one or more MigratoryData servers, subscribing to subjects, getting real-time messages for the subscribed subjects, and publishing real-time messages.

Public Member Functions

- MigratoryDataClient ()

  Create a MigratoryDataClient object.
- void SetLogListener (Object logListener, MigratoryDataLogType logType)

  Define a log listener object and configure the logging level.
- void SetListener (MigratoryDataListener listener)

  Attach a MigratoryDataListener for handling real-time messages and status notifications.
- void SetServers (string[ ] servers)

  Specify a cluster of one or more MigratoryData servers to which the client will connect to.
- void Subscribe (List< string > subjects)

  Subscribe to one or more subjects.
- void SubscribeWithConflation (List< string > subjects, int conflationTimeMillis)

  Subscribe to one or more subjects with conflation.
- void SubscribeWithHistory (List< string > subjects, int numberOfHistoricalMessages)

  Subscribe to one or more subjects after getting historical messages for those subjects.
- void Unsubscribe (List< string > subjects)

  Unsubscribe from one or more subjects.
- void SetEncryption (bool b)

  Configure whether to use SSL/TLS encryption when connecting to a MigratoryData server.
- void SetEntitlementToken (string token)

  Assign an authorization token to the client.
- List< string > GetSubjects ()

  Return the list of subscribed subjects.
- void SetServersDownBeforeNotify (int n)

  Define the number of failed attempts to connect to one or more MigratoryData servers before triggering a status notification MigratoryDataClient.NOTIFY_SERVER_DOWN.
- void Disconnect ()

  Disconnect from the connected MigratoryData server and dispose the resources used by the connection.

- void Publish (MigratoryDataMessage message)

  Publish a message.
- void NotifyAfterReconnectRetries (int retries)

  Define the number of failed attempts to connect to one or more MigratoryData servers before triggering a status notification MigratoryDataClient.NOTIFY_SERVER_DOWN.
- void SetQuickReconnectMaxRetries (int retries)

  Define the maximum number of retries for the Quick Reconnect failover phase.
- void SetQuickReconnectInitialDelay (int seconds)

  Define the number of seconds to wait before attempting to reconnect to the cluster after a connection failure is detected.
- void SetReconnectPolicy (string policy)

  Define the reconnect policy to be used after the Quick Reconnect phase.
- void setReconnectTimeInterval (int seconds)

  Define the time interval used for the reconnect schedule after the Quick Reconnect phase.
- void setReconnectMaxDelay (int seconds)

  Define the maximum reconnect delay for the MigratoryDataClient.TRUNCATED_EXPONENTIAL_BACKOFF policy.

Static Public Attributes

- static readonly string NOTIFY_SERVER_UP = "NOTIFY_SERVER_UP"

  Indicate that the client successfully connected to a MigratoryData server.
- static readonly string NOTIFY_SERVER_DOWN = "NOTIFY_SERVER_DOWN"

  Indicate that the client failed to connect to a MigratoryData server.
- static readonly string NOTIFY_DATA_SYNC = "NOTIFY_DATA_SYNC"

  After a failover reconnection, the client resynchronized a subscribed subject with the latest message available for that subject, as well as with all messages published during the failover for that subject.
- static readonly string NOTIFY_DATA_RESYNC = "NOTIFY_DATA_RESYNC"

  After a failover reconnection, the client resynchronized a subscribed subject with the latest message available for that subject, but not with the potential messages published during the failover, therefore behaving as a new client.
- static readonly string NOTIFY_SUBSCRIBE_ALLOW = "NOTIFY_SUBSCRIBE_ALLOW"

  Indicate that the client was authorized to subscribe to a subject.
- static readonly string NOTIFY_SUBSCRIBE_DENY = "NOTIFY_SUBSCRIBE_DENY"

  Indicate that the client was not authorized to subscribe to a subject.
- static readonly string **NOTIFY_SUBSCRIBE_TIMEOUT** = "NOTIFY_SUBSCRIBE_TIMEOUT"
- static readonly string NOTIFY_PUBLISH_OK = "NOTIFY_PUBLISH_OK"

  Indicate that the client successfully published a message.
- static readonly string NOTIFY_PUBLISH_FAILED = "NOTIFY_PUBLISH_FAILED"

  Indicate that the client was unable to publish a message.
- static readonly string NOTIFY_PUBLISH_DENIED = "NOTIFY_PUBLISH_DENIED"

  Indicate that the client was unable to publish a message because it is not allowed by the entitlement system.
- static readonly string NOTIFY_PUBLISH_NO_SUBSCRIBER = "NOTIFY_PUBLISH_NO_SUBSCRIBER"

  Indicate that the client was unable to publish a message because there is no client subscribed to the subject of the message.
- static readonly string CONSTANT_WINDOW_BACKOFF = "CONSTANT_WINDOW_BACKOFF"

  A constant used to define the reconnect policy.
- static readonly string TRUNCATED_EXPONENTIAL_BACKOFF = "TRUNCATED_EXPONENTIAL_BACK↩OFF"

  A constant used to define the reconnect policy.

4.1.1 Detailed Description

This class implements all the necessary operations for connecting to a cluster of one or more MigratoryData servers, subscribing to subjects, getting real-time messages for the subscribed subjects, and publishing real-time messages.

4.1.2 Member Function Documentation

4.1.2.1 SetLogListener()

```
void MigratoryDataClient.SetLogListener (
            Object logListener,
            MigratoryDataLogType logType )
```

Define a log listener object and configure the logging level.

The *log listener* object should belong to a class which implements the MigratoryDataLogListener interface.

It is advisable to configure this API call first if you want to log as much as possible.

Parameters

logListener	An instance of a class which implements the MigratoryDataLogListener interface
logType	The logging verbosity (MigratoryDataLogType.ERROR, MigratoryDataLogType.INFO, MigratoryDataLogType.DEBUG or MigratoryDataLogType.TRACE); by default there is no logging.

4.1.2.2 SetListener()

```
void MigratoryDataClient.SetListener (
            MigratoryDataListener listener )
```

Attach a MigratoryDataListener for handling real-time messages and status notifications.

Parameters

listener	An instance of a class which implements the MigratoryDataListener interface

4.1.2.3 SetServers()

```
void MigratoryDataClient.SetServers (
            string [] servers )
```

Specify a cluster of one or more MigratoryData servers to which the client will connect to.

If you specify two or more MigratoryData servers, then all these MigratoryData servers should provide the same level of data redundancy, by making available for subscription the same set of subjects. This is required for achieving (weighted) load balancing, failover, and guaranteed message delivery of the system. In this way, the MigratoryData servers of the `servers` list form a *cluster*.

For example, to connect to a cluster formed of two MigratoryData servers installed at the addresses `p1.example.com` and `p2.example.com`, and configured to accept connections on the standard HTTP port `80`, the following code can be used:

```
client.SetServers(new string[] {"p1.example.com:80", "p2.example.com:80"});
```

To achieve load-balancing, the API connects the client to a MigratoryData server chosen randomly from the `servers` list. In this way, the load is balanced among all the members of the cluster.

Moreover, the API supports weighted load-balancing. This feature is especially useful if the MigratoryData servers in the cluster are installed on machines with different capacities. You can assign to each member of the cluster a *weight* ranging from `0` to `100`. This weight assignment is a hint provided to the API to select with a higher probability a MigratoryData server with a higher weight either initially when the client connects to the cluster or later during a failover reconnection.

Supposing the address `p1.example.com` corresponds to a machine that is twice more powerful than the machine having the address `p2.example.com`, then you can assign to `p1.example.com` a weight `100` and to `p2.example.com` a weight `50` by prefixing each address with the assigned weight as follows:

```
client.SetServers(new string[] {"100 p1.example.com:80", "50 p2.example.com:80"});
```

The API assigns a default weight `100` to the addresses not prefixed with a specific weight.

To achieve failover, if the connection between the client and a MigratoryData server is broken, then the API will automatically detect the failure and will select another MigratoryData server from the `servers` list. If the client fails to connect to the new selected server, a status notification MigratoryDataClient.NOTIFY_SERVER_DOWN will be triggered (unless you modify the number of failed attempts with MigratoryDataClient.SetServersDownBeforeNotify()), and a new MigratoryData server in the cluster will be selected again and again until the client will be able to connect to one of the MigratoryData servers in the cluster. When successfully connected, the API will trigger a status notification MigratoryDataClient.NOTIFY_SERVER_UP.

Furthermore, if certified message delivery is enabled, then the potential messages published during the failover period for a subscribed subject, will be automatically retrieved from the cache of the MigratoryData server to which the client reconnects to and a status notification MigratoryDataClient.NOTIFY_DATA_SYNC will be triggered for that subject.

If, for example, the failover period is abnormally long, and the client is not able to retrieve, after a failover reconnection, the messages published during the failover period for one of its subscribed subjects, then the API will retrieve only the most recent message for that subject and will trigger a MigratoryDataClient.NOTIFY_DATA_RESYNC status notification for that subject, the client behaving as a new client which connects to the cluster at the moment of the failover reconnection.

For a complete discussion concerning the load balancing, failover, and certified message delivery features see the *MigratoryData Architecture Guide* (`PDF`, `HTML`).

Parameters

servers	An array of strings where each string represents the network address (IP address or DNS domain name and its corresponding port) of a MigratoryData server, optionally prefixed by a weight ranging from `0` to `100`. If the weight prefix is not provided to an address, then the API will automatically assign to that address a default weight `100`.

Exceptions

NullReferenceException	If the address of a MigratoryData server is null.
SocketException	If the connection to the address of a MigratoryData server could not be established.
FormatException	If the address of a MigratoryData server is not valid.

4.1.2.4 Subscribe()

```
void MigratoryDataClient.Subscribe (
            List< string > subjects )
```

Subscribe to one or more subjects.

Subscribe for real-time messages having as subjects the strings provided in the `subjects` parameter.

As an example, supposing the messages are market data updates having as subjects stock names. Then, to subscribe for the messages having as subjects `/stocks/NYSE/IBM` and `/stocks/Nasdaq/MSFT` the following code will be used:

```
List<string> subjects = new List<string>();
subjects.Add("/stocks/NYSE/IBM");
subjects.Add("/stocks/Nasdaq/MSFT");
client.Subscribe(subjects);
```

The subjects are strings having a particular particular syntax. See the Chapter "Concepts" in the *MigratoryData Architecture Guide* (PDF, HTML) to learn about the syntax of the subjects.

Parameters

subjects	An array of strings representing subjects.

4.1.2.5 SubscribeWithConflation()

```
void MigratoryDataClient.SubscribeWithConflation (
            List< string > subjects,
            int conflationTimeMillis )
```

Subscribe to one or more subjects with conflation.

Subscribe for real-time messages having as subjects the strings provided in the `subjects` parameter.

If the optional parameter `conflationTimeMillis` is used, then for each subject in the `subjects` list given in argument, its messages will be aggregated in the MigratoryData server and published every `conflationTimeMillis` milliseconds as aggregated data (containing only the latest value for that subject and its latest field values). The value of `conflationTimeMillis` should be a multiple of `100` milliseconds, otherwise the MigratoryData server will round it to the nearest value multiple of `100` milliseconds (e.g. `76` will be rounded to `0`, `130` will be

rounded to 100, 789 will be rounded to 700, ...). If the value of conflationTimeMillis is 0 (or is rounded to 0), then no conflation will apply, and data publication will be message-by-message with no message aggregation.

As an example, supposing the messages are market data updates having as subjects stock names. Then, to subscribe for the messages having as subjects /stocks/NYSE/IBM and /stocks/Nasdaq/MSFT using 1-second conflation the following code will be used:

```
List<string> subjects = new List<string>();
subjects.Add("/stocks/NYSE/IBM");
subjects.Add("/stocks/Nasdaq/MSFT");
client.SubscribeWithConflation(subjects, 1000);
```

The subjects are strings having a particular particular syntax. See the Chapter "Concepts" in the *MigratoryData Architecture Guide* (PDF, HTML) to learn about the syntax of the subjects.

Parameters

subjects	An array of strings representing subjects.
conflationTimeMillis	An optional argument defining the number of milliseconds used to aggregate ("conflate") the messages for each subject in the subjects list; default value is 0 meaning that no conflation will apply, and data publication will be message-by-message with no message aggregation.

4.1.2.6 SubscribeWithHistory()

```
void MigratoryDataClient.SubscribeWithHistory (
            List< string > subjects,
            int numberOfHistoricalMessages )
```

Subscribe to one or more subjects after getting historical messages for those subjects.

Attempt to get the number of historical messages as defined by the argument numberOfHistorical↩Messages, for each subject in the argument subjects, then subscribe for real-time messages having as subjects the strings provided in the subjects parameter.

When Guranteed Message Delivery is enabled, each MigratoryData server in the cluster maintains an in-memory cache with historical messages for each subject. The cache of each subject is available in all servers of the cluster. The maximum number of messages held in cache is defined by the parameter MaxCachedMessagesPer↩Subject of the MigratoryData server which defaults to 1,000 messages. The historical messages are continuously removed from the cache, but it is guaranteed that they are available in the cache at least the number of seconds defined by the parameter CacheExpireTime which defaults to 180 seconds.

If the value of numberOfHistoricalMessages is higher then the number of historical messages available in the cache, then the client will receive only the messages available in the cache. As a consequence, if you use a value higher than the value of the parameter MaxCachedMessagesPerSubject of the MigratoryData server (which defaults to 1000), then you will get the entire cache before subscribing for real-time messages for the subjects specified with the API call.

```
List<string> subjects = new List<string>();
subjects.Add("/stocks/NYSE/IBM");
subjects.Add("/stocks/Nasdaq/MSFT");
client.SubscribeWithHistory(subjects, 10);
```

The subjects are strings having a particular syntax. See the Chapter "Concepts" in the *MigratoryData Architecture Guide* (PDF, HTML) to learn about the syntax of the subjects.

Parameters

subjects	An array of strings representing subjects.
numberOfHistoricalMessages	The number of historical messages to be retrieved from the cache of the MigratoryData server. A value `0` means that no historical messages has to be retrieved and, in this case, this API method is equivalent to the API method MigratoryDataClient.Subscribe(). A value larger that the value of the parameter `MaxCachedMessagesPerSubject` means the entire cache is retrieved.

4.1.2.7 Unsubscribe()

```
void MigratoryDataClient.Unsubscribe (
            List< string > subjects )
```

Unsubscribe from one or more subjects.

Unsubscribe from the subscribed subjects provided in the `subjects` parameter.

Parameters

subjects	An array of strings representing subjects.

4.1.2.8 SetEncryption()

```
void MigratoryDataClient.SetEncryption (
            bool b )
```

Configure whether to use SSL/TLS encryption when connecting to a MigratoryData server.

When using encryption you have to connect to the ports of the MigratoryData servers that are configured to listen for encrypted connections. See the parameter `ListenEncrypted` in the *MigratoryData Configuration Guide* (PDF, HTML).

Parameters

b	Determine whether the client connects to the MigratoryData server using an encrypted SSL/TLS connection

4.1.2.9 SetEntitlementToken()

```
void MigratoryDataClient.SetEntitlementToken (
            string token )
```

Assign an authorization token to the client.

To define which users of your application have access to which subjects, you will first have to set the parameter `Entitlement` on `true` in the configuration file of the MigratoryData server - see the parameter `Entitlement` in the *MigratoryData Configuration Guide* (PDF, HTML).

Then, you will have to use the entitlement-related part of the MigratoryData Extension API to allow or deny certain users to subscribe to certain subjects.

Parameters

token	A string representing an authorization token.

4.1.2.10 GetSubjects()

```
List<string> MigratoryDataClient.GetSubjects ( )
```

Return the list of subscribed subjects.

Returns

A list of strings representing the subscribed subjects.

4.1.2.11 SetServersDownBeforeNotify()

```
void MigratoryDataClient.SetServersDownBeforeNotify (
            int n )
```

Define the number of failed attempts to connect to one or more MigratoryData servers before triggering a status notification MigratoryDataClient.NOTIFY_SERVER_DOWN.

Deprecated use NotifyAfterReconnectRetries

Parameters

n	The number of the failed attempts to connect to one or more MigratoryData servers before triggering a status notification MigratoryDataClient.NOTIFY_SERVER_DOWN; default value is `1`.

4.1.2.12 Disconnect()

```
void MigratoryDataClient.Disconnect ( )
```

Disconnect from the connected MigratoryData server and dispose the resources used by the connection.

This method should be called when the connection is no longer necessary.

4.1.2.13 Publish()

```
void MigratoryDataClient.Publish (
            MigratoryDataMessage message )
```

Publish a message.

If you attach a closure to the message, you will receive a status notification about this message publication via MigratoryDataListener.OnStatus().

Parameters

message	A MigratoryDataMessage message

4.1.2.14 NotifyAfterReconnectRetries()

```
void MigratoryDataClient.NotifyAfterReconnectRetries (
            int retries )
```

Define the number of failed attempts to connect to one or more MigratoryData servers before triggering a status notification MigratoryDataClient.NOTIFY_SERVER_DOWN.

Parameters

retries	The number of the failed attempts to connect to one or more MigratoryData servers before triggering a status notification MigratoryDataClient.NOTIFY_SERVER_DOWN; default value is `1`.

4.1.2.15 SetQuickReconnectMaxRetries()

```
void MigratoryDataClient.SetQuickReconnectMaxRetries (
            int retries )
```

Define the maximum number of retries for the Quick Reconnect failover phase.

Parameters

retries	The maximum number of quick reconnect retries; default value is `3`.

4.1.2.16 SetQuickReconnectInitialDelay()

```
void MigratoryDataClient.SetQuickReconnectInitialDelay (
            int seconds )
```

Define the number of seconds to wait before attempting to reconnect to the cluster after a connection failure is detected.

Connection Failure Detection

Connection failure is detected immediately for almost all users. For a few users which are subject to temporary, atypical network conditions, connection failure is detected after 30-40 seconds.

Reconnection Phases and Policies

When a connection failure is detected, the API will attempt to reconnect to the servers of the MigratoryData cluster as follows: First, it will attempt to reconnect up to a number of times as defined by MigratoryDataClient.SetQuick↩ ReconnectMaxRetries() using small delays between retries (Quick Reconnection Phase). If the connection cannot be established after the Quick Reconnection Phase, then the API will attempt to reconnect less frequently according to the policy defined by MigratoryDataClient.SetReconnectPolicy().

The delays between retries are computed according to the following algorithm where the values of the variables involved are defined by the API methods having substantially the same names:

```
Quick Reconnect Phase (retries <= quickReconnectMaxRetries)
------------------------------------------------------------

   (retries starts with 1 and increment by 1 at each quick reconnect)

   reconnectDelay = quickReconnectInitialDelay * retries - random(0, quickReconnectInitialDelay

After Quick Reconnect Phase (retries > quickReconnectMaxRetries)
-----------------------------------------------------------------

   (reset retries to start with 1 and increment by 1 at each reconnect)

   If reconnectPolicy is CONSTANT_WINDOW_BACKOFF, then

      reconnectDelay = reconnectTimeInterval

   else if reconnectPolicy is TRUNCATED_EXPONENTIAL_BACKOFF, then

      reconnectDelay = min(reconnectTimeInterval * (2 ^ retries) - random(0, reconnectTimeInter
```

For a few users which are subject to temporary, atypical network conditions, if `reconnectDelay` computed with the algorithm above is less than 10 seconds, then it is rounded to 10 seconds.

Parameters

seconds	The number of seconds to wait before attempting to reconnect to the cluster; default value is `5` seconds.

4.1.2.17 SetReconnectPolicy()

```
void MigratoryDataClient.SetReconnectPolicy (
            string policy )
```

Define the reconnect policy to be used after the Quick Reconnect phase.

See MigratoryDataClient.SetQuickReconnectInitialDelay() to learn about the Quick Reconnect phase and the reconnect schedule for the policy defined by this method.

Parameters

policy	The reconnect policy to be used after the Quick Reconnect phase. The possible values are MigratoryDataClient.CONSTANT_WINDOW_BACKOFF and MigratoryDataClient.TRUNCATED_EXPONENTIAL_BACKOFF; the default value is MigratoryDataClient.TRUNCATED_EXPONENTIAL_BACKOFF.

4.1.2.18 setReconnectTimeInterval()

```
void MigratoryDataClient.setReconnectTimeInterval (
            int seconds )
```

Define the time interval used for the reconnect schedule after the Quick Reconnect phase.

See MigratoryDataClient.SetQuickReconnectInitialDelay() to learn about the Quick Reconnect phase and how the value defined by this API method is used for the reconnect schedule.

Parameters

seconds	A time interval expressed in seconds used for reconnect schedule; default is `20` seconds.

4.1.2.19 setReconnectMaxDelay()

```
void MigratoryDataClient.setReconnectMaxDelay (
            int seconds )
```

Define the maximum reconnect delay for the MigratoryDataClient.TRUNCATED_EXPONENTIAL_BACKOFF policy.

See MigratoryDataClient.SetQuickReconnectInitialDelay() to learn how the value defined by this API method is used.

Parameters

seconds	The maximum reconnect delay when the policy MigratoryDataClient.TRUNCATED_EXPONENTIAL_BACKOFF is used; default value is 360 seconds.

4.1.3 Member Data Documentation

4.1.3.1 NOTIFY_SERVER_UP

```
readonly string MigratoryDataClient.NOTIFY_SERVER_UP = "NOTIFY_SERVER_UP"  [static]
```

Indicate that the client successfully connected to a MigratoryData server.

This constant indicates that the client successfully connected to one of the MigratoryData servers defined with the API method MigratoryDataClient.SetServers().

4.1.3.2 NOTIFY_SERVER_DOWN

```
readonly string MigratoryDataClient.NOTIFY_SERVER_DOWN = "NOTIFY_SERVER_DOWN"  [static]
```

Indicate that the client failed to connect to a MigratoryData server.

This constant indicates that the client failed to connect to one of the MigratoryData servers defined with the API method MigratoryDataClient.SetServers().

4.1.3.3 NOTIFY_DATA_SYNC

```
readonly string MigratoryDataClient.NOTIFY_DATA_SYNC = "NOTIFY_DATA_SYNC"  [static]
```

After a failover reconnection, the client resynchronized a subscribed subject with the latest message available for that subject, as well as with all messages published during the failover for that subject.

This constant indicates that the client successfully synchronized the subject provided in the detail information of the status notification. Also, the potential messages published for that subject during the failover period have been successfully retrieved at the moment of the reconnection.

4.1.3.4 NOTIFY_DATA_RESYNC

```
readonly string MigratoryDataClient.NOTIFY_DATA_RESYNC = "NOTIFY_DATA_RESYNC"  [static]
```

After a failover reconnection, the client resynchronized a subscribed subject with the latest message available for that subject, but not with the potential messages published during the failover, therefore behaving as a new client.

This constant indicates that the client successfully synchronized the subject provided in the detail information of the status notification. However, the client was unable to get the messages published during the failover. Therefore, it behaves like a new client which connects to the MigratoryData server at the moment of the failover reconnection.

4.1.3.5 NOTIFY_SUBSCRIBE_ALLOW

```
readonly string MigratoryDataClient.NOTIFY_SUBSCRIBE_ALLOW = "NOTIFY_SUBSCRIBE_ALLOW"  [static]
```

Indicate that the client was authorized to subscribe to a subject.

This constant indicates that the client identified with the token defined with the API method MigratoryDataClient.SetEntitlementToken() is allowed to subscribe to the subject provided in the detail information of the status notification.

4.1.3.6 NOTIFY_SUBSCRIBE_DENY

```
readonly string MigratoryDataClient.NOTIFY_SUBSCRIBE_DENY = "NOTIFY_SUBSCRIBE_DENY"  [static]
```

Indicate that the client was not authorized to subscribe to a subject.

This constant indicates that the client identified with the token defined with the API method MigratoryDataClient.SetEntitlementToken() is not allowed to subscribe to the subject provided in the detail information of the status notification.

4.1.3.7 NOTIFY_PUBLISH_OK

```
readonly string MigratoryDataClient.NOTIFY_PUBLISH_OK = "NOTIFY_PUBLISH_OK"  [static]
```

Indicate that the client successfully published a message.

This constant is used to indicate that the publication of the message, having the closure provided in the detail information of the status notification, has succeeded.

4.1.3.8 NOTIFY_PUBLISH_FAILED

```
readonly string MigratoryDataClient.NOTIFY_PUBLISH_FAILED = "NOTIFY_PUBLISH_FAILED"  [static]
```

Indicate that the client was unable to publish a message.

This constant is used to indicate that the publication of the message, having the closure provided in the detail information of the status notification, has failed.

4.1.3.9 NOTIFY_PUBLISH_DENIED

```
readonly string MigratoryDataClient.NOTIFY_PUBLISH_DENIED = "NOTIFY_PUBLISH_DENIED"  [static]
```

Indicate that the client was unable to publish a message because it is not allowed by the entitlement system.

This constant is used to indicate that the publication of the message, having the closure provided in the detail information of the status notification, has failed. The publication failed because the client identified with the token defined with the API method MigratoryDataClient.SetEntitlementToken() is not allowed to publish on the subject of the message.

4.1.3.10 NOTIFY_PUBLISH_NO_SUBSCRIBER

```
readonly string MigratoryDataClient.NOTIFY_PUBLISH_NO_SUBSCRIBER = "NOTIFY_PUBLISH_NO_SUBSCR↩
IBER"  [static]
```

Indicate that the client was unable to publish a message because there is no client subscribed to the subject of the message.

This constant is used to indicate that the publication of the message, having the closure provided in the detail information of the status notification, has failed. The publication failed because there is no client then subscribed to the subject of the message.

Deprecated no more is use.

4.1.3.11 CONSTANT_WINDOW_BACKOFF

```
readonly string MigratoryDataClient.CONSTANT_WINDOW_BACKOFF = "CONSTANT_WINDOW_BACKOFF"  [static]
```

A constant used to define the reconnect policy.

See MigratoryDataClient.SetQuickReconnectInitialDelay() for more details about this policy.

4.1.3.12 TRUNCATED_EXPONENTIAL_BACKOFF

```
readonly string MigratoryDataClient.TRUNCATED_EXPONENTIAL_BACKOFF = "TRUNCATED_EXPONENTIAL_B↩
ACKOFF"  [static]
```

A constant used to define the reconnect policy.

See MigratoryDataClient.SetQuickReconnectInitialDelay() for more details about this policy.

4.2 MigratoryDataField Class Reference

Represent a message field.

Public Member Functions

- MigratoryDataField (string name, string value)
  Create a MigratoryDataField object.

4.2.1 Detailed Description

Represent a message field.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 MigratoryDataField()

```
MigratoryDataField.MigratoryDataField (
            string name,
            string value )
```

Create a MigratoryDataField object.

Parameters

name	The field name
value	The field value

4.3 MigratoryDataListener Interface Reference

Implementations of this interface can handle the real-time messages received for the subscribed subjects as well as the status notifications.

Public Member Functions

- void OnMessage (MigratoryDataMessage message)

  This method handles the real-time messages received from the MigratoryData server for the subscribed subjects.
- void OnStatus (string status, string info)

  This method handles the status notifications.

4.3.1 Detailed Description

Implementations of this interface can handle the real-time messages received for the subscribed subjects as well as the status notifications.

Use the API method MigratoryDataClient.SetListener() to register your listener implementation.

4.3.2 Member Function Documentation

4.3.2.1 OnMessage()

```
void MigratoryDataListener.OnMessage (
            MigratoryDataMessage message )
```

This method handles the real-time messages received from the MigratoryData server for the subscribed subjects.

Parameters

message	An object of type MigratoryDataMessage .

4.3.2.2 OnStatus()

```
void MigratoryDataListener.OnStatus (
```

```
string status,
string info )
```

This method handles the status notifications.

The possible values of the `status` parameter are:

- `MigratoryDataClient.NOTIFY_SERVER_UP` indicates that the client successfully connected to the MigratoryData server provided in the detail information of the status notification
- `MigratoryDataClient.NOTIFY_SERVER_DOWN` indicates that the client was not able to connect to the MigratoryData server provided in the detail information of the status notification
- `MigratoryDataClient.NOTIFY_DATA_SYNC` indicates that, after a failover reconnection, the client successfully synchronized the subject given in the detail information of the status notification. Moreover, the client received the messages published during the failover period for this subject.
- `MigratoryDataClient.NOTIFY_DATA_RESYNC` indicates that, after a failover reconnection, the client successfully synchronized the subject given in the detail information of the status notification. However, the client have not received the potential messages published during the failover period for this subject, the client behaving like a new client which just connected to the MigratoryData server.
- `MigratoryDataClient.NOTIFY_SUBSCRIBE_ALLOW` indicates that the client identified with the token given in the argument of `MigratoryDataClient.SetEntitlementToken()` is allowed to subscribe to the subject provided in the detail information of the status notification
- `MigratoryDataClient.NOTIFY_SUBSCRIBE_DENY` indicates that the client identified with the token given in the argument of `MigratoryDataClient.SetEntitlementToken()` is not allowed to subscribe to the subject provided in the detail information of the status notification
- `MigratoryDataClient.NOTIFY_PUBLISH_OK` indicates that the client successfully published the message having the closure data provided in the detail information of the status notification
- `MigratoryDataClient.NOTIFY_PUBLISH_FAILED` indicates that the client was unable to publish the message having the closure data provided in the detail information of the status notification
- `MigratoryDataClient.NOTIFY_PUBLISH_DENIED` indicates that the client was unable to publish the message having the closure data provided in the detail information of the status notification because the client – identified with the token given in the argument of `MigratoryDataClient.SetEntitlementToken()` – is not allowed to publish on the subject of the message
- `MigratoryDataClient.NOTIFY_PUBLISH_NO_SUBSCRIBER` indicates that the client was unable to publish the message having the closure data provided in the detail information of the status notification because there is no client subscribed to the subject of the message

Parameters

status	The type of the status notification (see the possible values above).
info	The detail information of the status notification.

4.4 MigratoryDataLogListener Interface Reference

The listener interface that you should implement in order to get the logs of the API.

Public Member Functions

- void Error (string log)

  This method handles the MigratoryDataLogType.ERROR logs received from the API.
- void Info (string log)

  This method handles the MigratoryDataLogType.INFO logs received from the API.
- void Debug (string log)

  This method handles the MigratoryDataLogType.DEBUG logs received from the API.
- void Trace (string log)

  This method handles the MigratoryDataLogType.TRACE logs received from the API.

4.4.1 Detailed Description

The listener interface that you should implement in order to get the logs of the API.

Use the API method MigratoryDataClient.SetLogListener() to register your log listener implementation.

4.4.2 Member Function Documentation

4.4.2.1 Error()

```
void MigratoryDataLogListener.Error (
            string log )
```

This method handles the MigratoryDataLogType.ERROR logs received from the API.

Parameters

log	A string representing a MigratoryDataLogType.ERROR log.

4.4.2.2 Info()

```
void MigratoryDataLogListener.Info (
            string log )
```

This method handles the MigratoryDataLogType.INFO logs received from the API.

Parameters

log	A string representing a MigratoryDataLogType.INFO log.

4.4.2.3 Debug()

```
void MigratoryDataLogListener.Debug (
            string log )
```

This method handles the MigratoryDataLogType.DEBUG logs received from the API.

Parameters

log	A string representing a MigratoryDataLogType.DEBUG log.

4.4.2.4 Trace()

```
void MigratoryDataLogListener.Trace (
            string log )
```

This method handles the MigratoryDataLogType.TRACE logs received from the API.

Parameters

log	A string representing a MigratoryDataLogType.TRACE log.

4.5 MigratoryDataLogType Enum Reference

This class enumerates the MigratoryData logging levels.

Public Attributes

- ERROR

  The `ERROR` level turns on the error logs of the API.
- INFO

  The `INFO` level turns on the error and info logs of the API.
- DEBUG

  The `DEBUG` level turns on the error, info, and debug logs of the API.
- TRACE

  The `TRACE` level turns on all the logs of the API.

4.5.1 Detailed Description

This class enumerates the MigratoryData logging levels.

The available logging levels ordered by verbosity are:

- ERROR (less verbose)
- INFO
- DEBUG
- TRACE (most verbose)

For production usage, we recommend the `INFO` logging level.

4.6 MigratoryDataMessage Class Reference

Represent a message.

Public Member Functions

- MigratoryDataMessage (string subject, string content)
  Create a MigratoryDataMessage object.
- MigratoryDataMessage (string subject, string content, string closure)
  Create a MigratoryDataMessage object.
- MigratoryDataMessage (string subject, string content, List< MigratoryDataField > fields)
  Create a MigratoryDataMessage object.
- MigratoryDataMessage (string subject, string content, List< MigratoryDataField > fields, string closure)
  Create a MigratoryDataMessage object.

Protected Attributes

- bool **snapshot**
- bool **recovery**
- string **replyToSubject**
- int **seq**
- int **epoch**

4.6.1 Detailed Description

Represent a message.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 MigratoryDataMessage() [1/4]

```
MigratoryDataMessage.MigratoryDataMessage (
            string subject,
            string content )
```

Create a MigratoryDataMessage object.

Parameters

subject	The subject of the message
content	The content of the message

4.6.2.2 MigratoryDataMessage() [2/4]

```
MigratoryDataMessage.MigratoryDataMessage (
            string subject,
            string content,
            string closure )
```

Create a MigratoryDataMessage object.

Parameters

subject	The subject of the message
content	The content of the message
closure	The closure of the message

4.6.2.3 MigratoryDataMessage() [3/4]

```
MigratoryDataMessage.MigratoryDataMessage (
            string subject,
            string content,
            List< MigratoryDataField > fields )
```

Create a MigratoryDataMessage object.

Parameters

subject	The subject of the message
content	The content of the message
fields	The fields of the message

4.6.2.4 MigratoryDataMessage() [4/4]

```
MigratoryDataMessage.MigratoryDataMessage (
            string subject,
            string content,
```

```
        List< MigratoryDataField > fields,
        string closure )
```

Create a MigratoryDataMessage object.

Parameters

subject	The subject of the message
content	The content of the message
fields	The fields of the message
closure	The closure of the message

Index