

# **MigratoryData Client API for Java**

Developer's Guide and Reference Manual

*July 12, 2018*





# Contents

- 1 Developer's Guide** **1**
  
- 1.1 Overview 1
  
- 1.2 Creating Java clients for MigratoryData Server 1
  
- 1.2.1 Step 1 - Include the library 1
  
- 1.2.2 Step 2 - Define the listener for processing the real-time messages and status notifications 1
  
- 1.2.3 Step 3 - Specify the list of the MigratoryData servers where to connect to 2
  
- 1.2.4 Step 4 Subscribe to subjects and publish messages 2
  
- 1.2.5 Step 5 - Handle the real-time messages and status notifications 2
  
- 1.3 Examples 2
  
  
- 2 Deprecated List** **5**
  
  
- 3 Hierarchical Index** **7**
  
- 3.1 Class Hierarchy 7
  
  
- 4 Class Index** **9**
  
- 4.1 Class List 9

<b>5</b>	<b>Class Documentation</b>	<b>11</b>
5.1	MigratoryDataClient Class Reference	11
5.1.1	Detailed Description	12
5.1.2	Member Function Documentation	12
5.1.2.1	setLogging(MigratoryDataLogLevel logLevel, File logFile, int logRotateLimit)	12
5.1.2.2	setListener(MigratoryDataListener listener)	13
5.1.2.3	getListener()	13
5.1.2.4	setServers(String[] servers)	13
5.1.2.5	connect()	14
5.1.2.6	subscribe(List< String > subjects)	14
5.1.2.7	subscribeWithHistory(List< String > subjects, int numberOfHistoricalMessages)	15
5.1.2.8	unsubscribe(List< String > subjects)	15
5.1.2.9	setEncryption(boolean b)	16
5.1.2.10	setEntitlementToken(String token)	16
5.1.2.11	getSubjects()	16
5.1.2.12	setServersDownBeforeNotify(int n)	16
5.1.2.13	notifyAfterReconnectRetries(int retries)	17
5.1.2.14	disconnect()	17
5.1.2.15	publish(MigratoryDataMessage message)	17
5.1.2.16	setQuickReconnectMaxRetries(int retries)	17
5.1.2.17	setQuickReconnectInitialDelay(int seconds)	17
5.1.2.18	setReconnectPolicy(String policy)	18
5.1.2.19	setReconnectTimeInterval(int seconds)	19
5.1.2.20	setReconnectMaxDelay(int seconds)	19
5.2	MigratoryDataListener Interface Reference	19
5.2.1	Detailed Description	20
5.2.2	Member Function Documentation	21
5.2.2.1	onMessage(MigratoryDataMessage message)	21
5.2.2.2	onStatus(String status, String info)	21
5.2.3	Member Data Documentation	22

5.2.3.1	NOTIFY_SERVER_UP	22
5.2.3.2	NOTIFY_SERVER_DOWN	22
5.2.3.3	NOTIFY_DATA_SYNC	22
5.2.3.4	NOTIFY_DATA_RESYNC	22
5.2.3.5	NOTIFY_SUBSCRIBE_ALLOW	22
5.2.3.6	NOTIFY_SUBSCRIBE_DENY	23
5.2.3.7	NOTIFY_PUBLISH_OK	23
5.2.3.8	NOTIFY_PUBLISH_FAILED	23
5.2.3.9	NOTIFY_PUBLISH_DENIED	23
5.2.3.10	SERVICE_DESTROYED	23
5.2.3.11	SERVICE_STOPPED	24
5.2.3.12	SERVICE_DOWN_NO_NETWORK	24
5.2.3.13	SERVICE_START	24
5.2.3.14	SERVICE_RUNNING	24
5.2.3.15	INFO_ERROR_INVALID_SERVER	25
5.2.3.16	INFO_ERROR_INVALID_SUBJECT	25
5.2.3.17	CONSTANT_WINDOW_BACKOFF	25
5.2.3.18	TRUNCATED_EXPONENTIAL_BACKOFF	25
5.3	MigratoryDataLogLevel Enum Reference	25
5.3.1	Detailed Description	26
5.4	MigratoryDataMessage Class Reference	26
5.4.1	Detailed Description	27
5.4.2	Constructor & Destructor Documentation	27
5.4.2.1	MigratoryDataMessage(String subject, String content)	27
5.4.2.2	MigratoryDataMessage(String subject, String content, String closure)	27
5.4.3	Member Function Documentation	28
5.4.3.1	getSubject()	28
5.4.3.2	getContent()	28
5.4.3.3	getClosure()	28
5.4.3.4	isSnapshot()	28
5.4.3.5	setReplyToSubject(String subject)	28
5.4.3.6	getReplyToSubject()	29



# Chapter 1

## Developer's Guide

This guide includes the following sections:

- [Overview](#)
- [Creating Java clients for MigratoryData Server](#)
- [Examples](#)

### 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 Java clients for MigratoryData Server

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:

```
import com.migratorydata.client.*;
```

Also, include in the `class-path` of your application the API library `migratorydata-client-java.jar` located in the folder `lib` of this API package.

#### 1.2.2 Step 2 - 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 attach your listener implementation.

### 1.2.3 Step 3 - Specify the list of the 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 in order 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.

### 1.2.4 Step 4 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.

### 1.2.5 Step 5 - Handle the real-time messages and status notifications

Handle the messages received for the subscribed subjects as well as the status notifications in your listener implementation defined at Step 2 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.connect \(\)](#)**

same as using [MigratoryDataClient.setServers\(\)](#)

**Member [MigratoryDataClient.setServersDownBeforeNotify \(int n\)](#)**

use [notifyAfterReconnectRetries](#)

**Member [MigratoryDataListener.SERVICE\\_DESTROYED](#)**

no more in use



# Chapter 3

## Hierarchical Index

### 3.1 Class Hierarchy

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

- MigratoryDataClient . . . . . 11
- MigratoryDataListener . . . . . 19
- MigratoryDataLogLevel . . . . . 25
- Serializable
  - MigratoryDataMessage . . . . . 26



# Chapter 4

## Class Index

### 4.1 Class List

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

- [MigratoryDataClient](#)  
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 . . . . . 11
- [MigratoryDataListener](#)  
Implementations of this interface can handle the real-time messages received for the subscribed subjects as well as various status notifications . . . . . 19
- [MigratoryDataLogLevel](#)  
This class enumerates the MigratoryData logging levels . . . . . 25
- [MigratoryDataMessage](#)  
Represent a message . . . . . 26





# Chapter 5

## Class Documentation

### 5.1 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 [setLogging](#) ([MigratoryDataLogLevel](#) logLevel, File logFile, int logRotateLimit) throws IOException  
*Configure the logging parameters.*
- void [setListener](#) ([MigratoryDataListener](#) listener)  
*Attach a [MigratoryDataListener](#) for handling real-time messages and status notifications.*
- [MigratoryDataListener](#) [getListener](#) ()  
*Get the [MigratoryDataListener](#) object defined for handling real-time messages and status notifications.*
- void [setServers](#) (String[] servers) throws UnknownHostException  
*Specify a cluster of one or more MigratoryData servers to which the client will connect to.*
- void [connect](#) ()  
*Connect to a MigratoryData cluster.*
- void [subscribe](#) (List< String > subjects)  
*Subscribe to one or more subjects.*
- 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](#) (boolean 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.*
- Collection< 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 [MigratoryDataListener.NOTIFY\\_SERVER\\_DOWN](#).*

- 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 [disconnect](#) ()  
*Disconnect from the connected MigratoryData server and dispose the resources used by the connection.*
- void [publish](#) ([MigratoryDataMessage](#) message) throws Exception  
*Publish a message.*
- 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 [MigratoryDataListener.TRUNCATED\\_EXPONENTIAL\\_BACKOFF](#) policy.*

### 5.1.1 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.

### 5.1.2 Member Function Documentation

#### 5.1.2.1 void MigratoryDataClient.setLogging ( MigratoryDataLogLevel logLevel, File logFile, int logRotateLimit ) throws IOException

Configure the logging parameters.

It is advisable to configure this first if you want to log as much as possible. The default log level is [MigratoryDataLogLevel.INFO](#).

#### Parameters

<i>logLevel</i>	The particular <a href="#">MigratoryDataLogLevel</a> configured as the logging threshold
<i>logFile</i>	The file used to output the logs. For Android applications, set this parameter to <code>null</code> .
<i>logRotateLimit</i>	Define the maximum file size in bytes of the logging file to be used before creating a new logging file ( <i>log rotation</i> ). To disable log rotation, set this parameter on 0.

#### Exceptions

<i>IOException</i>	If there is an IO error while trying to configure the logging file.
--------------------	---

### 5.1.2.2 void MigratoryDataClient.setListener ( MigratoryDataListener listener )

Attach a [MigratoryDataListener](#) for handling real-time messages and status notifications.

#### Parameters

<i>listener</i>	An instance of a class which implements the <a href="#">MigratoryDataListener</a> interface
-----------------	---

### 5.1.2.3 MigratoryDataListener MigratoryDataClient.getListener ( )

Get the [MigratoryDataListener](#) object defined for handling real-time messages and status notifications.

#### Returns

The instance of a class which implements the [MigratoryDataListener](#) interface defined with [MigratoryDataClient.setListener\(\)](#)

### 5.1.2.4 void MigratoryDataClient.setServers ( String[] servers ) throws UnknownHostException

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 clients 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 `MigratoryDataListener.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 `MigratoryDataListener.NOTIFY_SERVER_UP`.

Furthermore, if guaranteed message delivery is enabled, then the potential messages published for a subscribed subject during the failover period, will be automatically retrieved from the cache of the MigratoryData server to which the client reconnects to and a status notification `MigratoryDataListener.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 available for that subject and will trigger a `MigratoryDataListener.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 related to load balancing, failover, and guaranteed message delivery features see the *MigratoryData Architecture Guide* ([PDF](#), [HTML](#)).

#### Parameters

<code>servers</code>	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

<code>UnknownHostException</code>	If the address of a MigratoryData server could not be determined
-----------------------------------	--

#### 5.1.2.5 void MigratoryDataClient.connect ( )

Connect to a MigratoryData cluster.

This API call can be used to connect to one of the MigratoryData servers specified with `MigratoryDataClient.setServers()`.

**Deprecated** same as using `MigratoryDataClient.setServers()`

#### 5.1.2.6 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 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 ArrayList<String>();
subjects.add("/stocks/NYSE/IBM");
subjects.add("/stocks/Nasdaq/MSFT");
client.subscribe(subjects);
```

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

<i>subjects</i>	An array of strings representing subjects.
-----------------	--

#### 5.1.2.7 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 `numberOfHistoricalMessages`, for each subject in the argument `subjects`, then subscribe for real-time messages having as subjects the strings provided in the `subjects` parameter.

When Guaranteed 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 `MaxCachedMessagesPerSubject` 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 than 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.

```
client.subscribeWithHistory(Arrays.asList("/stocks/NYSE/IBM", "/stocks/Nasdaq/MSFT"), 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

<i>subjects</i>	An array of strings representing subjects.
<i>numberOfHistoricalMessages</i>	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 <a href="#">MigratoryDataClient.subscribe()</a> . A value larger than the value of the parameter <code>MaxCachedMessagesPerSubject</code> means the entire cache is retrieved.

#### 5.1.2.8 void MigratoryDataClient.unsubscribe ( List< String > subjects )

Unsubscribe from one or more subjects.

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

#### Parameters

<code>subjects</code>	An array of strings representing subjects.
-----------------------	--

#### 5.1.2.9 void MigratoryDataClient.setEncryption ( boolean *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

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

#### 5.1.2.10 void MigratoryDataClient.setEntitlementToken ( String *token* )

Assign an authorization token to the client.

For more details about authorization see the parameter `Authorization.Type` in the *MigratoryData Configuration Guide* ([PDF](#), [HTML](#)).

#### Parameters

<code>token</code>	A string representing an authorization token.
--------------------	---

#### 5.1.2.11 Collection<String> MigratoryDataClient.getSubjects ( )

Return the list of subscribed subjects.

#### Returns

The list of strings representing the subscribed subjects.

#### 5.1.2.12 void MigratoryDataClient.setServersDownBeforeNotify ( int *n* )

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

**Deprecated** use `notifyAfterReconnectRetries`

## Parameters

<i>n</i>	The number of the failed attempts to connect to one or more MigratoryData servers before triggering a status notification <a href="#">MigratoryDataListener.NOTIFY_SERVER_DOWN</a> ; default value is 1.
----------	--

5.1.2.13 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

<i>retries</i>	The number of the failed attempts to connect to one or more MigratoryData servers before triggering a status notification <code>MigratoryDataClient.NOTIFY_SERVER_DOWN</code> ; default value is 1.
----------------	---

## 5.1.2.14 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.

5.1.2.15 void MigratoryDataClient.publish ( MigratoryDataMessage *message* ) throws Exception

Publish a message.

If the message includes a closure data, then a status notification will be provided via [MigratoryDataListener.onStatus\(\)](#) to inform whether the message publication has been successful or failed.

## Parameters

<i>message</i>	A <a href="#">MigratoryDataMessage</a> message
----------------	--

5.1.2.16 void MigratoryDataClient.setQuickReconnectMaxRetries ( int *retries* )

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

## Parameters

<i>retries</i>	The maximum number of quick reconnect retries; default value is 3.
----------------	--

5.1.2.17 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.setQuickReconnectMaxRetries()` 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

<i>seconds</i>	The number of seconds to wait before attempting to reconnect to the cluster; default value is 5 seconds.
----------------	--

### 5.1.2.18 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

<i>policy</i>	The reconnect policy to be used after the Quick Reconnect phase. The possible values are <a href="#">MigratoryDataListener.CONSTANT_WINDOW_BACKOFF</a> and <a href="#">MigratoryDataListener.TRUNCATED_EXPONENTIAL_BACKOFF</a> ; the default value is <a href="#">MigratoryDataListener.TRUNCATED_EXPONENTIAL_BACKOFF</a> .
---------------	---

#### 5.1.2.19 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

<i>seconds</i>	A time interval expressed in seconds used for reconnect schedule; default is 20 seconds.
----------------	--

#### 5.1.2.20 void MigratoryDataClient.setReconnectMaxDelay ( int seconds )

Define the maximum reconnect delay for the [MigratoryDataListener.TRUNCATED\\_EXPONENTIAL\\_BACKOFF](#) policy.

See [MigratoryDataClient.setQuickReconnectInitialDelay\(\)](#) to learn how the value defined by this API method is used.

#### Parameters

<i>seconds</i>	The maximum reconnect delay when the policy <a href="#">MigratoryDataListener.TRUNCATED_EXPONENTIAL_BACKOFF</a> is used; default value is 360 seconds.
----------------	--

## 5.2 MigratoryDataListener Interface Reference

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

### Public Member Functions

- void [onMessage](#) ([MigratoryDataMessage](#) message)  
*This method handles the real-time messages received from a MigratoryData server for the subscribed subjects.*
- void [onStatus](#) (String status, String info)  
*This method handles the status notifications.*

## Static Public Attributes

- static final String **NOTIFY\_SERVER\_UP** = "NOTIFY\_SERVER\_UP"  
*Indicate that the client successfully connected to a MigratoryData server.*
- static final String **NOTIFY\_SERVER\_DOWN** = "NOTIFY\_SERVER\_DOWN"  
*Indicate that the client failed to connect to a MigratoryData server.*
- static final String **NOTIFY\_DATA\_SYNC** = "NOTIFY\_DATA\_SYNC"  
*After a failover reconnection, the client synchronized 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 final String **NOTIFY\_DATA\_RESYNC** = "NOTIFY\_DATA\_RESYNC"  
*After a failover reconnection, the client synchronized 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 final String **NOTIFY\_SUBSCRIBE\_ALLOW** = "NOTIFY\_SUBSCRIBE\_ALLOW"  
*Indicate that the client was authorized to subscribe to a subject.*
- static final String **NOTIFY\_SUBSCRIBE\_DENY** = "NOTIFY\_SUBSCRIBE\_DENY"  
*Indicate that the client was not authorized to subscribe to a subject.*
- static final String **NOTIFY\_SUBSCRIBE\_TIMEOUT** = "NOTIFY\_SUBSCRIBE\_TIMEOUT"
- static final String **NOTIFY\_PUBLISH\_OK** = "NOTIFY\_PUBLISH\_OK"  
*Indicate that the client successfully published a message.*
- static final String **NOTIFY\_PUBLISH\_FAILED** = "NOTIFY\_PUBLISH\_FAILED"  
*Indicate that the client was unable to publish a message.*
- static final String **NOTIFY\_PUBLISH\_DENIED** = "NOTIFY\_PUBLISH\_DENIED"  
*Indicate that the client was unable to publish a message because it is not allowed by your entitlement rules.*
- static final String **SERVICE\_DESTROYED** = "SERVICE\_DESTROYED"  
*Indicate that the client was unable to publish a message because there is no client subscribed to the subject of the message.*
- static final String **SERVICE\_STOPPED** = "SERVICE\_STOPPED"  
*A constant holding the status type which indicates that the push notifications service has been stopped.*
- static final String **SERVICE\_DOWN\_NO\_NETWORK** = "SERVICE\_DOWN\_NO\_NETWORK"  
*A constant holding the status type which indicates that the push notifications service is down.*
- static final String **SERVICE\_START** = "SERVICE\_START"  
*A constant holding the status type which indicates that the push notifications service is up.*
- static final String **SERVICE\_RUNNING** = "SERVICE\_RUNNING"  
*A constant holding the status type which indicates that the push notifications service is running.*
- static final String **INFO\_ERROR\_INVALID\_SERVER** = "invalid server for client configuration"  
*A constant holding the info type which indicates that the push notification service has stopped because the list of servers provided with the method `MigratoryDataClient.setServers()` is null, empty or invalid. The info type is used with status notification `MigratoryDataListener.SERVICE_STOPPED`.*
- static final String **INFO\_ERROR\_INVALID\_SUBJECT** = "invalid subject for client configuration"  
*A constant holding the info type which indicates that the push notification service has stopped because the list of subjects is null, empty or invalid. The info type is used with status notification `MigratoryDataListener.SERVICE_STOPPED`.*
- static final String **CONSTANT\_WINDOW\_BACKOFF** = "CONSTANT\_WINDOW\_BACKOFF"  
*A constant used to define the reconnect policy.*
- static final String **TRUNCATED\_EXPONENTIAL\_BACKOFF** = "TRUNCATED\_EXPONENTIAL\_BACKOFF"  
*A constant used to define the reconnect policy.*

### 5.2.1 Detailed Description

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

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

## 5.2.2 Member Function Documentation

### 5.2.2.1 void MigratoryDataListener.onMessage ( MigratoryDataMessage message )

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

#### Parameters

<i>message</i>	An object of type <a href="#">MigratoryDataMessage</a> .
----------------	--

### 5.2.2.2 void MigratoryDataListener.onStatus ( String status, String info )

This method handles the status notifications.

The possible values of the `status` parameter are:

- [MigratoryDataListener.NOTIFY\\_SERVER\\_UP](#) indicates that the client successfully connected to the MigratoryData server provided in the detail information of the status notification
- [MigratoryDataListener.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
- [MigratoryDataListener.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.
- [MigratoryDataListener.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.
- [MigratoryDataListener.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
- [MigratoryDataListener.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
- [MigratoryDataListener.NOTIFY\\_PUBLISH\\_OK](#) indicates that the client successfully published the message having the closure data provided in the detail information of the status notification
- [MigratoryDataListener.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
- [MigratoryDataListener.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
- [MigratoryDataListener.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

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

### 5.2.3 Member Data Documentation

**5.2.3.1** `final String MigratoryDataListener.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\(\)](#).

**5.2.3.2** `final String MigratoryDataListener.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\(\)](#).

**5.2.3.3** `final String MigratoryDataListener.NOTIFY_DATA_SYNC = "NOTIFY_DATA_SYNC" [static]`

After a failover reconnection, the client synchronized 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.

**5.2.3.4** `final String MigratoryDataListener.NOTIFY_DATA_RESYNC = "NOTIFY_DATA_RESYNC" [static]`

After a failover reconnection, the client synchronized 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.

**5.2.3.5** `final String MigratoryDataListener.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.

5.2.3.6 `final String MigratoryDataListener.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.

5.2.3.7 `final String MigratoryDataListener.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.

5.2.3.8 `final String MigratoryDataListener.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.

5.2.3.9 `final String MigratoryDataListener.NOTIFY_PUBLISH_DENIED = "NOTIFY_PUBLISH_DENIED" [static]`

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

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.

5.2.3.10 `final String MigratoryDataListener.SERVICE_DESTROYED = "SERVICE_DESTROYED" [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 in use

A constant holding the status type which indicates that the Android service running the push notifications service has been destroyed.

This constant indicated that the Android service notification service has been killed by the operating system.

#### Attention

Used only in MigratoryData PushNotification API for Android

**5.2.3.11** `final String MigratoryDataListener.SERVICE_STOPPED = "SERVICE_STOPPED" [static]`

A constant holding the status type which indicates that the push notifications service has been stopped.

This constant indicated that the push notifications service is stopped.

**Attention**

Used only in MigratoryData PushNotification API for Android

**5.2.3.12** `final String MigratoryDataListener.SERVICE_DOWN_NO_NETWORK = "SERVICE_DOWN_NO_NETWORK" [static]`

A constant holding the status type which indicates that the push notifications service is down.

This constant indicated that the push notifications service has been stopped because the phone has no network connectivity.

**Attention**

Used only in MigratoryData PushNotification API for Android

**5.2.3.13** `final String MigratoryDataListener.SERVICE_START = "SERVICE_START" [static]`

A constant holding the status type which indicates that the push notifications service is up.

This constant indicated that the push notifications service has been started.

**Attention**

Used only in MigratoryData PushNotification API for Android

**5.2.3.14** `final String MigratoryDataListener.SERVICE_RUNNING = "SERVICE_RUNNING" [static]`

A constant holding the status type which indicates that the push notifications service is running.

This constant indicated that the push notifications service is running.

**Attention**

Used only in MigratoryData PushNotification API for Android

```
5.2.3.15 final String MigratoryDataListener.INFO_ERROR_INVALID_SERVER = "invalid server for client configuration"
        [static]
```

A constant holding the info type which indicates that the push notification service has stopped because the list of servers provided with the method [MigratoryDataClient.setServers\(\)](#) is null, empty or invalid. The info type is used with status notification [MigratoryDataListener.SERVICE\\_STOPPED](#) .

This constant indicated that the list of servers is invalid.

#### Attention

Used only in MigratoryData PushNotification API for Android

```
5.2.3.16 final String MigratoryDataListener.INFO_ERROR_INVALID_SUBJECT = "invalid subject for client configuration"
        [static]
```

A constant holding the info type which indicates that the push notification service has stopped because the list of subjects is null, empty or invalid. The info type is used with status notification [MigratoryDataListener.SERVICE\\_STOPPED](#) .

This constant indicated that the list of subjects is invalid.

#### Attention

Used only in MigratoryData PushNotification API for Android

```
5.2.3.17 final String MigratoryDataListener.CONSTANT_WINDOW_BACKOFF = "CONSTANT_WINDOW_BACKOFF"
        [static]
```

A constant used to define the reconnect policy.

See [MigratoryDataClient.setQuickReconnectInitialDelay\(\)](#) for more details about this policy.

```
5.2.3.18 final String MigratoryDataListener.TRUNCATED_EXPONENTIAL_BACKOFF = "TRUNCATED_EXPONENTIAL_BACKOFF"
        [static]
```

A constant used to define the reconnect policy.

See [MigratoryDataClient.setQuickReconnectInitialDelay\(\)](#) for more details about this policy.

## 5.3 MigratoryDataLogLevel Enum Reference

This class enumerates the MigratoryData logging levels.

## Public Attributes

- **TRACE**

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

- **DEBUG**

*The DEBUG level turns on the debug, info, warning, and error logs of the API.*

- **INFO**

*The INFO level turns on the info, warning, and error logs of the API.*

- **WARN**

*The WARN level turns on the warning and error logs of the API.*

- **ERROR**

*The ERROR level turns on the error logs of the API.*

### 5.3.1 Detailed Description

This class enumerates the MigratoryData logging levels.

The available logging levels ordered by verbosity are:

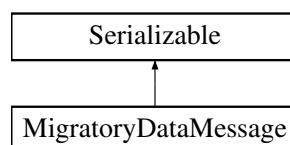
- ERROR (less verbose)
- WARN
- INFO
- DEBUG
- TRACE (most verbose)

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

## 5.4 MigratoryDataMessage Class Reference

Represent a message.

Inheritance diagram for MigratoryDataMessage:





## Public Member Functions

- [MigratoryDataMessage](#) (String subject, String content)  
*Create a [MigratoryDataMessage](#) object.*
- [MigratoryDataMessage](#) (String subject, String content, String closure)  
*Create a [MigratoryDataMessage](#) object.*
- String [getSubject](#) ()  
*Get the subject of the message.*
- String [getContent](#) ()  
*Get the content of the message.*
- String [getClosure](#) ()  
*Get the closure of the message.*
- boolean [isSnapshot](#) ()  
*Test whether the message is a snapshot message or not.*
- void [setReplyToSubject](#) (String subject)
- String [getReplyToSubject](#) ()
- String [toString](#) ()  
*Return a string representation of the message.*

## Protected Attributes

- boolean **isSnapshot**
- boolean **isRecovery**
- int **seq**
- int **epoch**

### 5.4.1 Detailed Description

Represent a message.

### 5.4.2 Constructor & Destructor Documentation

#### 5.4.2.1 MigratoryDataMessage.MigratoryDataMessage ( String *subject*, String *content* )

Create a [MigratoryDataMessage](#) object.

##### Parameters

<i>subject</i>	The subject of the message
<i>content</i>	The content of the message

#### 5.4.2.2 MigratoryDataMessage.MigratoryDataMessage ( String *subject*, String *content*, String *closure* )

Create a [MigratoryDataMessage](#) object.

## Parameters

<i>subject</i>	The subject of the message
<i>content</i>	The content of the message
<i>closure</i>	The closure of the message

### 5.4.3 Member Function Documentation

#### 5.4.3.1 String MigratoryDataMessage.getSubject ( )

Get the subject of the message.

**Returns**

A string representing the subject of the message

#### 5.4.3.2 String MigratoryDataMessage.getContent ( )

Get the content of the message.

**Returns**

A string representing the content of the message

#### 5.4.3.3 String MigratoryDataMessage.getClosure ( )

Get the closure of the message.

**Returns**

The closure data of the message

#### 5.4.3.4 boolean MigratoryDataMessage.isSnapshot ( )

Test whether the message is a snapshot message or not.

**Returns**

true if the message is a snapshot message

#### 5.4.3.5 void MigratoryDataMessage.setReplyToSubject ( String *subject* )

Set the subject to be used to reply to this message.

If a reply subject is attached to a message with this method, the message acts as a request. The clients which receive a request message will be able to reply by sending a message having as subject the reply subject.

If the reply subject is not already subscribed, it is subscribed by the API library implicitly. It can be reused for subsequent request/reply interactions (and even for receiving multiple replies to one request). When it is not needed anymore, it should be unsubscribed explicitly.

**Parameters**

<i>subject</i>	The subject to be used to reply to this message.
----------------	--

**5.4.3.6 String MigratoryDataMessage.getReplyToSubject ( )**

Get the subject to be used to reply to this message.

A client which receives a message containing a reply subject should interpret the message as a request. It has the option to use the reply subject - extracted from the message with this method - to send a reply.

**Returns**

The subject to be used to reply to this message.

# Index

- CONSTANT\_WINDOW\_BACKOFF
  - MigratoryDataListener, 25
- connect
  - MigratoryDataClient, 14
- disconnect
  - MigratoryDataClient, 17
- getClosure
  - MigratoryDataMessage, 28
- getContent
  - MigratoryDataMessage, 28
- getListener
  - MigratoryDataClient, 13
- getReplyToSubject
  - MigratoryDataMessage, 29
- getSubject
  - MigratoryDataMessage, 28
- getSubjects
  - MigratoryDataClient, 16
- INFO\_ERROR\_INVALID\_SERVER
  - MigratoryDataListener, 24
- INFO\_ERROR\_INVALID\_SUBJECT
  - MigratoryDataListener, 25
- isSnapshot
  - MigratoryDataMessage, 28
- MigratoryDataClient, 11
  - connect, 14
  - disconnect, 17
  - getListener, 13
  - getSubjects, 16
  - notifyAfterReconnectRetries, 17
  - publish, 17
  - setEncryption, 16
  - setEntitlementToken, 16
  - setListener, 12
  - setLogging, 12
  - setQuickReconnectInitialDelay, 17
  - setQuickReconnectMaxRetries, 17
  - setReconnectMaxDelay, 19
  - setReconnectPolicy, 18
  - setReconnectTimeInterval, 19
  - setServers, 13
  - setServersDownBeforeNotify, 16
  - subscribe, 14
  - subscribeWithHistory, 15
  - unsubscribe, 15
- MigratoryDataListener, 19
  - CONSTANT\_WINDOW\_BACKOFF, 25
  - INFO\_ERROR\_INVALID\_SERVER, 24
  - INFO\_ERROR\_INVALID\_SUBJECT, 25
  - NOTIFY\_DATA\_RESYNC, 22
  - NOTIFY\_DATA\_SYNC, 22
  - NOTIFY\_PUBLISH\_DENIED, 23
  - NOTIFY\_PUBLISH\_FAILED, 23
  - NOTIFY\_PUBLISH\_OK, 23
  - NOTIFY\_SERVER\_DOWN, 22
  - NOTIFY\_SERVER\_UP, 22
  - NOTIFY\_SUBSCRIBE\_ALLOW, 22
  - NOTIFY\_SUBSCRIBE\_DENY, 22
  - onMessage, 21
  - onStatus, 21
  - SERVICE\_DESTROYED, 23
  - SERVICE\_DOWN\_NO\_NETWORK, 24
  - SERVICE\_RUNNING, 24
  - SERVICE\_START, 24
  - SERVICE\_STOPPED, 23
  - TRUNCATED\_EXPONENTIAL\_BACKOFF, 25
- MigratoryDataLogLevel, 25
- MigratoryDataMessage, 26
  - getClosure, 28
  - getContent, 28
  - getReplyToSubject, 29
  - getSubject, 28
  - isSnapshot, 28
  - MigratoryDataMessage, 27
  - setReplyToSubject, 28
- NOTIFY\_DATA\_RESYNC
  - MigratoryDataListener, 22
- NOTIFY\_DATA\_SYNC
  - MigratoryDataListener, 22
- NOTIFY\_PUBLISH\_DENIED
  - MigratoryDataListener, 23
- NOTIFY\_PUBLISH\_FAILED
  - MigratoryDataListener, 23
- NOTIFY\_PUBLISH\_OK
  - MigratoryDataListener, 23
- NOTIFY\_SERVER\_DOWN
  - MigratoryDataListener, 22
- NOTIFY\_SERVER\_UP
  - MigratoryDataListener, 22
- NOTIFY\_SUBSCRIBE\_ALLOW
  - MigratoryDataListener, 22
- NOTIFY\_SUBSCRIBE\_DENY
  - MigratoryDataListener, 22
- notifyAfterReconnectRetries
  - MigratoryDataClient, 17

onMessage  
    MigratoryDataListener, 21

onStatus  
    MigratoryDataListener, 21

publish  
    MigratoryDataClient, 17

SERVICE\_DESTROYED  
    MigratoryDataListener, 23

SERVICE\_DOWN\_NO\_NETWORK  
    MigratoryDataListener, 24

SERVICE\_RUNNING  
    MigratoryDataListener, 24

SERVICE\_START  
    MigratoryDataListener, 24

SERVICE\_STOPPED  
    MigratoryDataListener, 23

setEncryption  
    MigratoryDataClient, 16

setEntitlementToken  
    MigratoryDataClient, 16

setListener  
    MigratoryDataClient, 12

setLogging  
    MigratoryDataClient, 12

setQuickReconnectInitialDelay  
    MigratoryDataClient, 17

setQuickReconnectMaxRetries  
    MigratoryDataClient, 17

setReconnectMaxDelay  
    MigratoryDataClient, 19

setReconnectPolicy  
    MigratoryDataClient, 18

setReconnectTimeInterval  
    MigratoryDataClient, 19

setReplyToSubject  
    MigratoryDataMessage, 28

setServers  
    MigratoryDataClient, 13

setServersDownBeforeNotify  
    MigratoryDataClient, 16

subscribe  
    MigratoryDataClient, 14

subscribeWithHistory  
    MigratoryDataClient, 15

TRUNCATED\_EXPONENTIAL\_BACKOFF  
    MigratoryDataListener, 25

unsubscribe  
    MigratoryDataClient, 15