QuantHouse® FeedOS™

FeedOS™ Developer's Notice

Oslo Børs Data Feed Migration to MillenniumIT

Reference n°: 20121031

Effective as of: 12 November 2012

Action required from users: Mandatory

QuantHouse® FeedOS™
FeedOS™ Developer's Notice
Reference 20121031
October 31, 2012

Corporate Headquarters
QuantHouse®
52 Rue de la Victoire
75009 Paris
France
Tel: +33 (0) 1 73 02 32 11
Fax: +33 (0) 1 73 02 32 12

UK Office
10 Foster Lane
London EC2V 6HR
United Kingdom
Tel: +44 (0) 203 107 1676

US Offices
55 Water Street, 44th floor
New York, NY 10041
United States of America
Tel: +1-(212)-438-4346

130 East Randolph
One Prudential Plaza, Suite 2900
Chicago, IL 60601
United States of America
Tel: +1-(312)-233-7129

www.quanthouse.com

OSLO BØRS DATA FEED MIGRATION TO MILLENNIUMIT

To reflect the changes caused by the migration of the Oslo Børs market data stream to the MillenniumIT format, QuantHouse® has decided to enhance the content of QuantFEED®. These changes also require customers using Oslo Børs Data Feed for replay purposes to upgrade their FeedOS™ API.

This developer's notice contains late-breaking information about the implementation of this modification in your applications, which may not be included otherwise in the published documentation. The topics this notice covers include:

- 1. Update Summary
- 2. Functional Description
- 3. QuantFEED® Technical Implementation
- 4. MIC Update
- 5. Upgrading FeedOS™ API for Replay Purposes
- 6. Finding the Latest Information.

1. Update Summary

Table 1 Current update summary

Notice Reference	20121031
Scope	Reference Data
Exchanges	Oslo Børs – Oslo MIT Data Feed
Effective Date	**2012-11-12**
Impact	• Update of the Referential and Quotation Tags • Update of the MICs • FeedOS™ API Upgrade for Feed Replay
Action required	Mandatory

2. Functional Description

Starting Monday, **November 12, 2012**, QuantHouse® changes the content of the referential tag `LocalCodeStr` (**NumericID:** `9500`, **Type:** `String`) to accommodate the information disseminated on Oslo Børs, following the migration to the MillenniumIT format.

Moreover, the new quotation tags `MARKET_LSE_MIT_TradingStatusDetails` (**NumericID:** `14750`, **Type:** `String`) and `MARKET_LSE_MIT_HaltReason` (**NumericID:** `14752`, **Type:** `String`) will be disseminated in QuantFEED®'s Level 1 Data Stream to detail the trading status and halt reason, respectively.

The new quotation context tag `MARKET_LSE_MIT_AuctionTypeIndicator` (**NumericID:** `15951`, **Type:** `String`) will be disseminated in QuantFEED®'s Level 1 Data Stream to detail the auction type.

Furthermore, the quotation tag `TradingStatus` (**NumericID:** `9100`, **Type:** `Enum`) has different possible values.

3. QuantFEED® Technical Implementation

The following sections describe the technical implementation of the new or updated quotation tags:

- 3.1. Local Code
- 3.2. Trading Status Details
- 3.3. Halt Reason
- 3.4. Auction Type Indicator
- 3.5. Trading Status.

3.1. Local Code

The values of the referential tag **Local Code** conveyed on the Oslo Børs market data stream are disseminated via QuantHouse®'s data stream in *Referential* to detail the local code of an instrument.

QuantFEED® implementation of the tag `LocalCodeStr` is described in the table below:

Table 2 LocalCodeStr – technical implementation in QuantFEED®

Component	Value	Description
Tag Name	`LocalCodeStr`	QuantFEED® tag name.
Numeric ID	`9500`	QuantFEED® unique ID broadcast on QuantHouse®'s data stream. This is the numeric equivalent of the tag name.
Type	`String`	String data type.
Format / Possible Values	*`[Exchange Specific Value]`*	An ***exchange specific value***, detailing the local code of an instrument.

3.2. Trading Status Details

Each time a modification of the instrument status occurs, the values of the quotation tag **Trading Status Details** conveyed on the Oslo Børs market data stream are disseminated via QuantHouse®'s data stream in *Other Values* to indicate the current status of the instrument:

- in the callback carrying the Level1 event `notif_TradeEventExt()`, for C++
- in the event handler `TradeEventExtEventHandler`, for C#
- in the callback carrying the Level1 event `quotNotifTradeEventExt`, for Java.

QuantFEED® implementation of the tag `MARKET_LSE_MIT_TradingStatusDetails` is described in the table below:

Table 3 MARKET_LSE_MIT_TradingStatusDetails – technical implementation in QuantFEED®

Component	Value	Description
Tag Name	`MARKET_LSE_MIT_TradingStatusDetails`	QuantFEED® tag name.
Numeric ID	`14750`	QuantFEED® unique ID broadcast on QuantHouse®'s data stream. This is the numeric equivalent of the tag name.
Type	`String`	String data type.
Format	*`[Exchange Specific Value]`*	An ***exchange specific value***, detailing the current status of an instrument.
Possible Values	`H`	Halt
	`T`	Regular Trading/Start of Trade Reporting
	`a`	Opening Auction Call
	`b`	Post-Close
	`c`	Market Closed (Close)
	`d`	Closing Auction Call
	`e`	Re-Opening Auction Call
	`v`	End of Trade Reporting
	`w`	No Active Session
	`x`	End of Post Close
	`y`	Pre-Trading (Start of Trading)
	`Z`	Closing Price Publication

3.3. Halt Reason

Each time an instrument is halted from trading, the values of the quotation tag **Halt Reason** conveyed on the Oslo Børs market data stream are disseminated via QuantHouse®'s data stream in *Other Values*:

- in the callback carrying the Level1 event `notif_TradeEventExt()`, for C++
- in the event handler `TradeEventExtEventHandler`, for C#
- in the callback carrying the Level1 event `quotNotifTradeEventExt`, for Java.

QuantFEED® implementation of the tag `MARKET_LSE_MIT_HaltReason` is described in the table below:

Table 4 MARKET_LSE_MIT_HaltReason – technical implementation in QuantFEED®

Component	Value	Description
Tag Name	`MARKET_LSE_MIT_HaltReason`	QuantFEED® tag name.
Numeric ID	`14752`	QuantFEED® unique ID broadcast on QuantHouse®'s data stream. This is the numeric equivalent of the tag name.
Type	`String`	String data type.
Format	*`[Exchange Specific Value]`*	An ***exchange specific value***, detailing the reason of halting an instrument.

Table 4 MARKET_LSE_MIT_HaltReason – technical implementation in QuantFEED® (Continued)

Component	Value	Description
Possible Values	1	Price movement
	2	Received announcement
	3	In anticipation of announcement
	4	System problems
	5	Other
	6	Reference data update
	101	Instrument-level circuit breaker tripped
	9998	Matching partition suspended
	9999	System suspended
	Space	Reason not available

When an instrument is no longer halted, the tag `MARKET_LSE_MIT_HaltReason` is reset by sending a value with the syntax `UNKNOWN`, as described in the example below:

```
FeedOS::Types::ListOfQuotationVariable const & values = inData.getValues();
for (FeedOS::Types::ListOfQuotationVariable::const_iterator it = values.begin()
; it != values.end(); ++it)
{
   unsigned int tag_num = it->getNum();
   if (tag_num == FeedOS::TAG_MARKET_LSE_MIT_HaltReason)
   {
      Any halt_reason_value = it->getValue();
      if (halt_reason_value.get_syntax() == Syntax_UNKNOWN)
      {
          // instrument is no more "halted": reset HaltReason
      }
      else
      {
         std::string reason_code = halt_reason_value.get_String();
         // process/store HaltReason
      }
      break;
   }
}
```

3.4. Auction Type Indicator

The values of the quotation tag **Auction Type Indicator** conveyed on the Oslo Børs market data stream are disseminated via QuantHouse®'s data stream in *Context* to detail the auction type:

- in the callback carrying the Level1 event `notif_TradeEventExt()`, for C++
- in the event handler `TradeEventExtEventHandler`, for C#
- in the callback carrying the Level1 event `quotNotifTradeEventExt`, for Java.

QuantFEED® implementation of the tag `MARKET_LSE_MIT_AuctionTypeIndicator` is described in the table below:

Table 5 MARKET_LSE_MIT_AuctionTypeIndicator – technical implementation in QuantFEED®

Component	Value	Description
Tag Name	`MARKET_LSE_MIT_AuctionTypeIndicator`	QuantFEED® tag name.
Numeric ID	`15951`	QuantFEED® unique ID broadcast on QuantHouse®'s data stream. This is the numeric equivalent of the tag name.
Type	`String`	String data type.
Format	*`[Exchange Specific Value]`*	An ***exchange specific value***, detailing the auction type.
Possible Values	`A`	Re-opening Auction
	`C`	Closing Auction
	`O`	Opening Auction

3.5. Trading Status

Each time a modification of the trading status occurs, the values of the quotation tag **Trading Status** conveyed on the Oslo Børs market data stream are disseminated via QuantFEED®'s data stream in *Other Values*:

- in the callback carrying the Level1 event `notif_TradeEventExt()`, for C++
- in the event handler `TradeEventExtEventHandler`, for C#
- in the callback carrying the Level1 event `quotNotifTradeEventExt`, for Java.

QuantFEED®'s implementation of the tag `TradingStatus` is described in the following table:

Table 6 TradingStatus – technical implementation in QuantFEED®

Component	Value	Description
Tag Name	`TradingStatus`	QuantFEED® tag name.
Numeric ID	`9100`	QuantFEED® unique ID disseminated on QuantHouse®'s data stream. This is the numeric equivalent of the tag name.
Type	`Enum`	Enum data type.
Format	*`[Exchange Specific Value]`*	An ***exchange specific value***, detailing the characteristics of the trading status.
Possible Values	`2`	Trading Halt
	`5`	Price Indication
	`15`	New Price Indication
	`17`	Ready to Trade
	`18`	Not Available for Trading

4. MIC Update

Starting Monday, **November 12, 2012**, QuantHouse® also adds two new MICs to accommodate the information disseminated on Oslo Børs market data stream. Thus, if the market segment is `OABM`, then the new MIC is `XOAM`; if the market segment is `OAX`, then the new MIC is `XOAS`. For the other segments, the MIC remains the same – `XOSL`.

Moreover, the **Internal Code** and the **Local Code** of the instruments whose MIC changes will also change.

The table below summarizes the new MIC distribution, corresponding to each market segment (new MICs are in **bold**):

Table 7 Correspondence between market segment and MIC

Market Segment	MIC
OABM	**XOAM**
OABT	XOSL
OAX	**XOAS**
OBAU	XOSL
OBBA	XOSL
OBEF	XOSL
OBEN	XOSL
OBIA	XOSL
OBMA	XOSL
OBOC	XOSL
OBPC	XOSL
OBST	XOSL
OBTM	XOSL
OBTS	XOSL
OBUB	XOSL
OBWR	XOSL
OBX	XOSL

5. Upgrading FeedOS™ API for Replay Purposes

To be able to replay the Oslo Børs Data Feed that is recorded after the migration date – **November 12, 2012** –, you should upgrade the FeedOS™ API to the minimum required version, as described in the table below:

Table 8 Currently required version to replay Oslo Børs Data Feed

Language	FeedOS™ API – minimum required version
C++	3.6.3.3
C#	2.4.3.4

For more details about the upgrade procedure, see *FeedOS™API Guide*.

6. Finding the Latest Information

For the latest documentation and product updates, additional support and training, please contact our support services:

- E-mail: support@quanthouse.com
- Web: http://support.quanthouse.com.