FIX Market Data Service

Technical Specification

v1.0.6

2025-09-16

Bucharest Stock Exchange

<dezvoltare-it@bvb.ro>

Table of Contents

Version History

Version	Date	Description
1.0.0	2016-11-16	Initial document release
1.0.1	2024-01-16	Add transparency trade flag in MarketDataIncrementalRefresh [X] with MDEntryTypes(269) = Trade(2) Update data dictionary to FIX 5.0 SP2 EP283
1.0.2	2024-02-14	Custom tag 5902 and 5903 format: UTC time only(HH:mm:ss)
1.0.3	2024-03-22	Add messages samples
1.0.4	2024-07-01	Add Administrative Messages for change password; update Custom tag section
1.0.5	2024-08-12	Add Settle Price and Open Interest snapshot and subscription.
1.0.6	2025-09-16	Add Bandwidth recommendation

Introduction

Purpose

This document contains a conceptual overview of the FIX protocol and provides technical guidance on adopting FIX to connect to Bucharest Stock Exchange and receive market data.

Bucharest Stock Exchange provides FIX connectivity for versions 5.0 SP2 EP283 of the FIX protocol

This document is intended to supplement the FIX Protocol Specification (www.fixprotocol.org). It is assumed that the reader is familiar with the FIX protocol as described at http://www.fixprotocol.org.

In this document FIX is a synonym for FIX protocol. And Exchange is a synonym for Bucharest Stock Exchange

What is FIX?

The Financial Information Exchange (FIX) Protocol is a messaging standard developed specifically for the real-time electronic Exchange of securities transactions. FIX is a public-domain specification owned and maintained by FIX Protocol, Ltd.

For further information, see the FIX specification: http://www.fixprotocol.org

Session management

Counterparty Identification - SenderCompID, TargetCompID

Client CompIDs are defined by the Exchange (as the user code) and can have up to 10 alphanumeric characters. Client CompIDs are case-sensitive. Client CompIDs could be bound to an IP address.

Clients should identify themselves in the SenderCompID(49) and should insert the Exchange identifier in the TargetCompID(56). The client will receive messages with SenderCompID(49) filled with the Exchange identifier and TargetCompID(56) filed with his own identifier.

One client can have only one open session at a time. Note then OnBefalfOf functionality is not used.

Session Life

Message sequence numbers are always reset to 1, when the FIX client connects.

Recovery after disconnections

The FIX session mechanism based on ResendRequest is not used. Message sequence numbers are always reset to 1 on Logon. In case of disconnect, Client has the possibility to request a gap filing for a certain period of time using MarketDataRequest[V]. Only trades and index values are subject to gap filling. All other information (bid, offer, imbalance, symbol status) are available only with snapshots and updates.

Custom Fields

Table 1. Custom Fields

Tag	Field Name	Field Type	Used in messages	Description
5902	Gap filing start	UTC time only (HH:mm:ss)	MarketDataRequest[V]	to get trades or indexes between two timestamps.

Tag	Field Name	Field Type	Used in messages	Description
5903	Gap filing end	UTC time only (HH:mm:ss)	MarketDataRequest[V]	to get trades or indexes between two timestamps.
1300	MarketSegmentID	STRING	SecurityList[y]	The primary Exchange market code where the instrument is traded
893	LastFragment	boolean	SecurityStatus[f]	Y = Mark the end of response stream
336	TradingSessionID	STRING	SecurityStatus[f]	Add custom values: <value enum="O" description="(Continuous) Trading"/> <value enum="C" description="CLOSED"/> <value enum="F" description="FIXING"/> <value enum="PO" description="PREOPENED"/> <value enum="PC" description="PRECLOSED"/> <value enum="TAL" description="TRADING_AT_LAST"/> <value enum="POX" description="PREOPENED_EXTENDED"/> <value enum="PCX" description="PRECLOSED_EXTENDED"/> <value enum="V" description="VOLATILITY INTERRUPTION"/> <value enum="VX" description="VOLATILITY INTERRUPTION EXTENDED"/>

Message Header format and Standard Trailer

Table 2. Message Header

Tag	Field Name	Req	Description
8	BeginString	Y	BeginString
9	BodyLength	Y	Must be the second field in the message.
34	MsgSeqNum	Y	See standard FIX explanation.

Tag	Field Name	Req	Description
35	MsgType	Y	Must be the third field in the message.
43	PossDupFlag	N	Always required for retransmissions.
49	SenderCompID	Y	The value must be recognized and agreed to by Exchange
52	SendingTime	Y	Indicates the time the message was sent by the client.
56	TargetCompID	Y	The value must be recognized and agreed to by Exchange
1137	DefaultApplVerID	N	Specifies the service pack release being applied, by default, to message at the session level. Supported values: 7= FIX50

Each message is terminated by a standard trailer as per FIX protocol.

Session messages

Logon

The first messages exchanged in a FIX session are the Logon Request and the Logon Response. The Logon Request is sent by the client, and will be followed by the Logon Response sent by the Exchange. A logon request can be reated using the structure above, also adding the password tag '554'. An example can be found at the end of the documentation. If the password is expired, a business reject will be received with the message 'Password expired, change your password'. Also, if the password is expired, Market Data request will return the same message until the password is changed.

A client has to wait for a Logon Response before starting to send business messages to the Exchange.

The Logon message always resets the session sequence to 1. Also note that any `ResendRequest` will be rejected. In case you want to recover business data (only trades and index values) you will be using a custom tagged `MarketDataRequest[V]` (more details to follow) for which you will receive a stream of `MarketDataIncrementalRefresh[X]` also with some custom tags in order to signal the end of stream.

Logout

FIX clients should terminate a session by logging out.

Administrative Messages (change password)

User Request Message[BE]

User Request is used to change the password of the user.

Table 3. UserRequestMessage[BE]

Tag	Field Name	Value
923	UserRequestID	Required by FIX; ignored by the EXCHANGE
924	UserRequest Type	3= Change Password For User
553	UserName	User code (case sensitive)
554	Password	Old Password
925	NewPassword	New password
Component	Standard Trailer	Y

User Response Message[BF]

```
The EXCHANGE responds to a UserRequest with an UserResponse as per FIX protocol.
Note that a User Response does not have UserRequestID filled with the original one
supplied in UserRequest.
```

Table 4. UserResponseMessage[BF]

Tag	Field Name	Value
923	UserRequestID	Required by FIX; ignored by the EXCHANGE
553	UserName	User code (case sensitive)
926	UserStatus	Supported values: 5 = Password Changed 6 = Other (password not changed)
927	UserStatustText	Description of error
Component	Standard Trailer	Y

Application messages

Instrument identification

Instruments are identified within the Exchange by their symbol code. At the same time an instrument can be traded in more than one market each having an exchange given code. Clients will receive in the Symbol(55) tag a `symbol.market` combination that will identify a concrete order-

book within the Exchange. For example the order-book `SIF1.REGS` identifies the symbol `SIF1` traded in the `REGS` market.

Clients must subscribe in order to receive order-book and index data. The Symbol(55) tag specified in MarketDataRequest[V] will be used as a criteria to select which order-book/index the client wants to receive data about. For order-books one can put a market code in the Symbol(55) tag in order to receive data about all instruments traded in that market. For indices the Symbol(55) tag will contain the index code. If the client wants to subscribe to all order-books/indices then a single Symbol(55) entry with value * can be used.

Note that is not possible to use a single MarketDataRequest[V] to subscribe to both order-book and index data. One must use two MarketDataRequest[V] in order to do that, one for order-books, one for index data.

The next example shows a subscription for all instruments traded on REGS and RGSP markets:

Table 5. MarketDataRequest[V] for order book and trade subscription

Tag	Field Name	Value
262	MDReqID	12345
263	SubscriptionRequestType	1 [Snapshot + Updates (Subscribe)]
264	MarketDepth	0
265	MDUpdateType	1 [Incremental refresh]
266	AggregatedBook	Y [book entries to be aggregated]
146	NoRelatedSym	2
155	Symbol	REGS
155	Symbol	RGSP
267	NoMDEntryTypes	4
269	MDEntryTypes	0 [Bid]
269	MDEntryTypes	1 [Offer]
269	MDEntryTypes	2 [Trade]
269	MDEntryTypes	A [Imbalance]
Component	Standard Trailer	Y

The next example shows a subscription for all indices:

Table 6. MarketDataRequest[V] for Index value subscribe

Tag	Field Name	Value
262	MDReqID	12345
263	SubscriptionRequestType	1 [Snapshot + Updates (Subscribe)]
264	MarketDepth	0
265	MDUpdateType	1 [Incremental refresh]

Tag	Field Name	Value
266	AggregatedBook	Y [book entries to be aggregated]
146	NoRelatedSym	1
155	Symbol	*
267	NoMDEntryTypes	1
269	MDEntryTypes	3 [Index]
Component	Standard Trailer	Y

Message flow

Table 7. Message flow

Use Case	Incoming Message In FIXMD service	Incoming Message Detail	Outgoing message from FIXMD Service	Outgoing message detail
Securities availabl e in trading system	SecurityListReque st[x]		SecurityList[y]	a single message will contain all the securities
Securiti es status	SecurityStatusReq uest[e]	SubscriptionReque stType(263) possible values: 0 = Snapshot 1 = Snapshot + Updates (Subscribe)	SecurityStatus[f]	One message for each security; LastFragment(893) is present with having Y as value for the last message of snaphot; If SubscriptionRequestType(26 3) = Snapshot + Updates (1) then snapshot is followed by subsequent updates in the security status

Use Case	Incoming Message In FIXMD service	Incoming Message Detail	Outgoing message from FIXMD Service	Outgoing message detail
Order book - Snapshot and Updates	MarketDataRequest[V]	SubscriptionRequestType(263) 1 = Snapshot + Updates (Subscribe) MDEntryTypes(269) one of 0 = Bid 1 = Offer	MarketDataSnapshotFullRefresh[W] for snapshot followed by MarketDataIncrementalRefresh[X] for next updates	MarketDataSnapshotFullRefresh[W] will have MDReqID(262) equal with the request id supplied in MarketDataRequest[V]; there will be a special entry marking the end of the snapshot response stream having Symbol(55)=* and MDEntryType(269)=J(Empty order book) Any update in order book will be supplied with message MarketDataIncrementalRefresh[X]
Trades Gap - Trades executed between two specified timestamps	MarketDataRequest[V]	SubscriptionRequestType(263) 0 = Snapshot MDEntryTypes(269) 2= Trade Custom tag 5902 = start time (HH:mm:ss) Custom tag 5903 = end time (HH:mm:ss)	MarketDataIncrementalRefresh[X]	MDReqID(262) will be equal with the request id supplied in MarketDataRequest[V]; MDEntryType(269)=J(Empty order book) will mark the end of the response stream
Trade - Updates	MarketDataRequest[V]	SubscriptionRequestType(263) 1 = Snapshot + Updates (Subscribe) MDEntryTypes(269) 2 = Trade	MarketDataIncrementalRefresh[X]	

Use Case	Incoming Message In FIXMD service	Incoming Message Detail	Outgoing message from FIXMD Service	Outgoing message detail
Index Gap - index values between two specified timestamps	MarketDataRequest[V]	SubscriptionRequestType(263) 0 = Snapshot MDEntryTypes(269) 3 = Index Custom tag 5902 = start time (HH:mm:ss) Custom tag 5903 = end time (HH:mm:ss)	MarketDataIncrementalRefresh[X]	MDReqID(262) will be equal with the request id supplied in MarketDataRequest[V] MDEntryType(269)= J (Empty order book) will mark the end of the response stream
Index - Snapshot and Updates	MarketDataRequest[V]	SubscriptionRequestType(263) 1 = Snapshot + Updates (Subscribe) MDEntryTypes(269) 3 = Index	MarketDataSnapshotFullRefresh[W] followed by MarketDataIncrementalRefresh[X]	MarketDataSnapshotFullRefresh[W] will have MDReqID(262) equal with the request id supplied in MarketDataRequest[V]; there will be a special entry marking the end of response stream having Symbol(55)=* and MDEntryType(269)=J(Empty order book)
Settlement Price Snapshot	MarketDataRequest[V]	SubscriptionRequestType(263) 0 = Snapshot MDEntryType(269) 6 = Settle Price	MarketDataSnapshotFullRefresh[W]	MDReqID(262) will be equal with the request id supplied in MarketDataRequest[V]; MDEntryType(269)=J(Empty order book) will mark the end of the response stream
Settlement Price Update	MarketDataRequest[V]	SubscriptionRequestType(263) 1 = Snapshot + Updates (Subscribe) MDEntryType(269) 6 = Settle Price	MarketDataIncrementalRefresh[X]	

Use Case	Incoming Message In FIXMD service	Incoming Message Detail	Outgoing message from FIXMD Service	Outgoing message detail
Open Interest Snapshot	MarketDataRequest[V]	SubscriptionRequestType(263) 0 = Snapshot MDEntryType(269) C = Open Interest	MarketDataSnapshotFullRefresh[W]	MDReqID(262) will be equal with the request id supplied in MarketDataRequest[V]; MDEntryType(269)=J(Empty order book) will mark the end of the response stream
Open Interest Update	MarketDataRequest[V]	SubscriptionRequestType(263) 1 = Snapshot + Updates (Subscribe) MDEntryType(269) C = Open Interest	MarketDataIncrementalRefresh[X]	

SecurityListRequest[x]

Table 8. SecurityListRequest

Tag	Field Name	Req	Description
Component	Standard Header	Y	
320	SecurityReqID	Y	Unique ID of a Security Definition Request
559	SecurityListRequestType	Y	4=All Securities
Component	Standard Trailer	Y	

SecurityList[y]

Table 9. SecurityList

Tag	Field Name	Req	Description
Component	Standard Header	Y	
320	SecuritytReqID	Y	Unique ID of a Security Request
Component	SecListGrp	N	
146	NoRelatedSym	Y	
55	Symbol	N	The symbol code given by the Exchange
107	SecurityDesc	N	The name of the instrument
22	SecurityIDSource	N	4 = ISIN Number
48	SecurityID	N	Instrument ISIN

Tag	Field Name	Req	Description
15	Currency	N	Trading currency
461	CFICode	N	CFI Code
423	PriceType	N	1=Percentage 2=Per unit 9=Yield
1300	MarketSegmentID	N	The primary Exchange market code where the instrument is traded
893	LastFragment	N	Y = Last message
Component	Standard Trailer	Y	

SecurityStatusRequest[e]

Table 10. SecurityStatusRequest

Tag	Field Name	Req	Description
Component	Standard Header	Y	
324	SecurityStatusReqID	Y	
263	SubscriptionRequestType	Y	1 = Snapshot + Updates (Subscribe) 2 = Disable previous Snapshot + Update Request (Unsubscribe)
55	Symbol	Y	* = all Exchange markets [market code] = a specific Exchange market code
Component	Standard Trailer	Y	

SecurityStatus[f]

Table 11. SecurityStatus

Tag	Field Name	Req	Description
Component	Standard Header	Y	
324	SecurityStatusReqID	N	Available if this is a response to a snapshot request
55	Symbol	N	The Exchange `symbol.market` combination

Tag	Field Name	Req	Description
336	TradingSessionID	N	O = Opened C = Closed F = Fixing - uncrossing PO = Preopen PC = Preclose TAL = Trading at last POX = Preopen extended PCX = Preclose extended V = Volatility interruption VX = Volatility interruption extended
60	TransactTime	N	The UTC timestamp when this entity changed
893	LastFragment	N	Y = Last message; available to mark the end of response stream
Component	Standard Trailer	Y	

MarketDataRequest[V]

Table 12. MarketDataRequest[V]

Tag	Field Name	Req	Description
Component	Standard Header	Y	
262	MDReqID	Y	Unique identifier for Market Data Request Required if SubscriptionRequestType(263) = 2(Disable previous Snapshot + Updates Request).
263	SubscriptionRequestType	Y	0 = Snapshot 1 = Snapshot + Updates (Subscribe) 2 = Disable previous Snapshot + Update Request (Unsubscribe)
264	MarketDepth	Y	0 = full book depth
265	MDUpdateType	Y	1 = Incremental refresh

266	AggregatedBook	Y	Y = book entries to be aggregated
Component	MDReqGroup	Y	
267	NoMDEntryTypes	Y	Number of MDEntryType (269) fields requested
269	MDEntryType	Y	Accepted values: If SubscriptionRequestType(263) = 0(Snapshot) 2 = Trade 3 = Index value 6 = Settle Price C = Open Interest If SubscriptionRequestType(263) = 1(Subscribe) 0 = Bid 1 = Offer 2 = Trade 3 = Index value 6 = Settle Price A = Imbalance C = Open Interest Note: if either Bid(0) or Offer(1) is specified then Bid, Offer and Imbalance data will be present in MDIncrementaRefresh.
Component	InstrmtMDReqGrp	Y	
146	NoRelatedSym	Y	
55	Symbol	Y	Exchange market code or * for all markets; if MDEntryType = 3 then this field must be an index code or * for all indices
5902	Gap filing start	N	Use in conjunction with SubscriptionRequestType(263)=0. Format HH:mm:ss. Absence of this field is interpreted as begin of the day.

5903	Gap filing end	N	Use in conjunction with SubscriptionRequestType(263)=0. Format HH:mm:ss. Absence of this field is interpreted as current time.
Component	Standard Trailer	Y	

MarketDataIncrementalRefresh[X]

Table 13. MarketDataIncrementalRefresh[X]

Tag	Field Name	Req	Description
Component	Standard Header	Y	
Component	MDIncGrp	Y	
268	NoMDEntries	Y	Number of entries following.
279	MDUpdateAction	Y	Supported values: 0 = New 1 = Change 2 = Delete
269	MDEntryType	Y	Valid values: 0 = Bid 1 = Offer 2 = Trade 3 = Index value 6 = Settle Price A = Imbalance C = Open Interest
278	MDEntryID	N	available only if MDEntryType(269) = Trade(2); contains the unique trade identifier as given by the Exchange

40	OrdType	N	available only if MDEntryType(269) = Bid(0) or Offer(1); valid values: K = Market With Left Over as Limit 2 = Limit
270	MDEntryPx	N	price for entry or index value
271	MDEntrySize	N	volume for entry; not available if MDEntryType(269) = 3(Index)
272	MDEntryDate	N	Date of Market Data Entry
273	MDEntryTime	N	UTC Timestamp of entry
1024	MDOriginType	N	available only if MDEntryType(269) = Trade(2); Valid values: 0 = Book 3 = Quote driven market 4 = Dark order book 1 = Off-book 5 = Auction driven market 6 = Quote negotation 8 = Hybrid market

625	TradingSessionSubID	N	available only if MDEntryType(269) = Trade(2); Valid values: 8 = Any auction 2 = Opening or opening auction 4 = Closing or closing auction 6 = Intraday auction 9 = Unscheduled intraday auction 3 = (Continuous) trading 5 = Post-trading 10 = Out of main session trading
574	MatchType	N	available only if MDEntryType(269) = Trade(2); Valid values: 3 = Confirmed trade report (reporting from recognized markets) 1 = One Party Trade Report (privately negotiated trade) 9 = Systematic Internalizer
1115	OrderCategory	N	available only if MDEntryType(269) = Trade(2); Valid values: 3 = Privately negotiated trade
2669	TrdRegPublicationType	N	available only if MDEntryType(269) = Trade(2); Valid values: 0 = Pre-trade transparency waiver

2670	TrdRegPublication Reason	N	available only if MDEntryType(269) = Trade(2); Valid values: 0 = No preceding order in book as transaction price set within average spread of a liquid instrument) 1 = No preceding order in book as transaction price depends on system-set reference price for an illiquid instrument) 2 = No preceding order in book as transaction price is subject to conditions other than current market price 4 = No public price quoted as instrument is illiquid
829	TrdRegPublication Reason	N	available only if MDEntryType(269) = Trade(2); Valid values: 37 = Crossed trade
2667	AlgorithmicTradeIndicator	N	available only if MDEntryType(269) = Trade(2); Valid values: 1 = Algorithmic trade 0 = Non-algorithmic trade
Component	Instrument	Y	
55	Symbol	Y	instrument identifier as an Exchange `symbol.market` combination; if MDEntryType(269) = Index(3) this value will contain the index code
Component	Standard Trailer	Y	

MarketDataSnapshotFullRefresh[W]

Table 14. MarketDataSnapshotFullRefresh[W]

Tag	Field Name	Req	Description
Component	Standard Header	Y	
262	MDReqID	Y	the id of MarketDataRequest[V]

1021	MDBookType	N	Supported values: 1 = Top of Book if MDEntryType(269) = 3(Index value) 2 = Price Depth
Component	Instrument	Y	
55	Symbol	Y	- instrument identifier as an Exchange `symbol.market` combination - if MDEntryType(269) = Index(3) this will contain the index code - it will be * if this entry signifies the end of the response stream (in this case MDEntryType(269) will be J(Empty book))
Component	MDFullGrp	Y	
268	NoMDEntries	Y	Number of entries in Market Data message.
269	MDEntryType	N	Supported values: 0 = Bid 1 = Offer 3 = Index value 6 = Settle Price A = Imbalance C = Open Interest J = Empty book
270	MDEntryPx	N	price for entry or index value
271	MDEntrySize	N	volume for entry; not available if MDEntryType(269) = 3(Index)
272	MDEntryDate	N	Date of Market Data Entry
273	MDEntryTime	N	UTC Timestamp of entry

1024	MDOriginType	N	available only if MDEntryType(269) = Trade(2); Valid values: 0 = Book 3 = Quote driven market 4 = Dark order book 1 = Off-book 5 = Auction driven market 6 = Quote negotation 8 = Hybrid market
625	TradingSessionSubID	N	available only if MDEntryType(269) = Trade(2); Valid values: 8 = Any auction 2 = Opening or opening auction 4 = Closing or closing auction 6 = Intraday auction 9 = Unscheduled intraday auction 3 = (Continuous) trading 5 = Post-trading 10 = Out of main session trading

574	MatchType	N	available only if MDEntryType(269) = Trade(2); Valid values: 3 = Confirmed trade report (reporting from recognized markets) 1 = One Party Trade Report (privately negotiated trade) 9 = Systematic Internalizer
1115	OrderCategory	N	available only if MDEntryType(269) = Trade(2); Valid values: 3 = Privately negotiated trade
2669	TrdRegPublicationType	N	available only if MDEntryType(269) = Trade(2); Valid values: 0 = Pre-trade transparency waiver
2670	TrdRegPublicationReason	N	available only if MDEntryType(269) = Trade(2); Valid values: 0 = No preceding order in book as transaction price set within average spread of a liquid instrument) 1 = No preceding order in book as transaction price depends on system-set reference price for an illiquid instrument) 2 = No preceding order in book as transaction price is subject to conditions other than current market price 4 = No public price quoted as instrument is illiquid
829	TrdRegPublicationReason	N	available only if MDEntryType(269) = Trade(2); Valid values: 37 = Crossed trade

2667	AlgorithmicTradeIndicator	N	available only if MDEntryType(269) = Trade(2); Valid values: 1 = Algorithmic trade 0 = Non-algorithmic trade
Component	Standard Trailer	Y	

MarketDataRequestReject[Y]

Table 15. MarketDataReject[Y]

Tag	Field Name	Req	Description
Component	Standard Header	Y	
262	MDReqID	Y	MDReqID from request
281	MDReqRejReason	N	
58	Text	N	free text
Component	Standard Trailer	Y	

Reconstructing the order-book

Since we are sending incremental refresh updates the order-book state must be reconstructed at the client side:

- start with a MarketDataRequest[V] in order to obtain a MarketDataSnapshotFullRefresh[W] which will be the snapshot upon which the following MarketDataIncrementalRefresh[X] will be applied
- build two lists, one for each side (MDEntryType) and sort the entries according to their trading priority in each list:
  - for instruments having PriceType(423)=1(Percentage) or 2(Per unit), buy entries must be sorted descending based on their price (a higher price means a higher priority), sell levels must be sorted ascending based on their price (a lower price means a higher priority)
  - for instruments having PriceType(423)=9(Yield) it's the other way around i.e. for buy entries a lower price means a higher priority and for sell entries a higher price means a higher priority
  - a price entry having OrdType(40)=K(Market With Left Over as Limit) has the maximum priority no matter its' side (MDEntryType) or instrument's PriceType
- if a MarketDataIncrementalRefresh[X] arrives having entries with MDUpdateAction(279)=0(New) and prices that do not exist in the current order-book the client must create new price entries and insert them into the order-book based on their trading priority (see above)

- if a MarketDataIncrementalRefresh[X] arrives having entries with MDUpdateAction(279)=0(New) and prices that already exists in the current order-book the client must update the price entry by adding the incoming MDEntrySize to the existing accumulated size on that price level
- if a MarketDataIncrementalRefresh[X] arrives having entries with MDUpdateAction(279)=2(Delete) and prices that already exist in the current order-book the client must update the price entry by subtracting the incoming MDEntrySize from the existing accumulated size on that price level; if the resulting size becomes zero the entire price level must be removed from the order-book
- if a MarketDataIncrementalRefresh[X] arrives having entries with MDUpdateAction(279)=2(Delete) and prices that do not exist in the current order-book the client must drop the update; however it this happens it must be considered a bug and the entire order-book is to be rebuilt from scratch; please notify us if this happens.
- the client must apply a MarketDataIncrementalRefresh[X] atomically on the current order-book and processing the update entries in the order they appear in the message
- if the client receives MarketDataIncrementalRefresh[X] messages before MarketDataSnapshotFullRefresh[W] arrives they should be dropped

Settle Price and Open Interest Subscriptions

It is also possible to request a snapshot or subscribe to 'Settle Price' (MDEntryType(269) = '6') and 'Open Interest' (MDEntryType(269) = 'C') for all the instruments from the exchange. 'Open Interest' is only applicable for instruments of type 'FUTURES'. In this case, symbols will be identified by the symbol ticker from the exchange (e.g. BVB, H2O, TLV etc). A snapshot request will return the data from today.
For Market Data Requests of type Snapshot, updates of type MarketDataSnapshotFullRefresh will be received. The end of this request will be market with MDEntryType(269) = 'J'.
In the case of subscription, the updates will be of type MarketDataIncrementalRefresh.
Examples are shown below.

Messages Examples

Logon

Request example

```
8=FIXT.1.1 9=90 35=A 34=1 49=USER01 52=20240807-14:19:59.520 56=BUX 98=0 108=30 141=Y
554=******** 1137=7 10=126
```

Response example on successful attempt

```
8=FIXT.1.1 9=83 35=A 34=1 49=BUX 52=20240807-14:19:59.591 56=USER01 1128=7 98=0 108=30
```

```
141=Y 1137=7 10=107
```

SecurityListRequest

Request example:

```
8=FIXT.1.1 9=78 35=x 34=605 49=RAZVAN 52=20240228-16:34:32.917 56=BUX
320=1709138072917 559=4 10=051
```

Example of a Simplified Response, Security List:

```
8=FIXT.1.1 9=168607 35=y 34=1160 49=BUX 52=20240228-16:34:33.213 56=RAZVAN 1128=7
320=1709138072917 893=Y 146=1998 55=TALD 48=ROTALDACNOR9 22=4 107=TALC DOLOMITA SA
HUNEDOARA 423=2 15=RON 1300=XRS1 55=ATPA 48=ROATPAACNOR6 22=4 107=ATHENEE PALACE
BUCURESTI 423=2 15=RON 1300=XRS1 55=MOBE 48=ROMOBEACNOR4 22=4 107=MOBEST BUCURESTI
423=2 15=RON 1300=XRS1 55=SCBC 48=ROSCBCACNOR0 22=4 107=SCUT BACAU 423=2 15=RON
1300=XRS1 55=MINX 48=ROMINXACNOR1 22=4 107=MINEXFOR DEVA 423=2 15=RON 1300=XRS1
55=BNAT 48=ROBNATACNOR9 22=4 107=BANAT ESTIVAL 2002 OLIMP 423=2 15=RON 1300=XRS1
55=EBSNGTL1 48=AT0000A13JF8 22=4 107=EB SNG TURBO LONG L1 423=2 15=RON 1300=RGSP
10=225
```

MarketDataIncrementalRefresh[X]

Examples: MarketDataIncrementalRefresh[X]

```
8=FIXT.1.1 9=141 35=X 34=1496 49=BUX 52=20240228-16:48:09.930 56=RAZVAN 1128=7 268=1
279=2 269=0 55=IMP.REGS 270=0.237 271=5000 272=20240228 273=16:48:09.928 10=124
```

MDUpdateAction (tag:279) = 2 (DELETE)

MDEntryType (tag:269) = 0 (BID)

MarketDataIncrementalRefresh[X]

```
8=FIXT.1.1 9=141 35=X 34=1497 49=BUX 52=20240228-16:48:09.941 56=RAZVAN 1128=7 268=1
279=0 269=1 55=IMP.REGS 270=0.237 271=5000 272=20240228 273=16:48:09.938 10=127
```

MDUpdateAction (tag:279) = 0 (NEW)

MDEntryType (tag:269) = 1 (OFFER)

MarketDataRequest[V]

Request example of MarketDataRequest[V] Snapshot for BET index:

```
8=FIXT.1.1 9=109 35=V 34=659 49=RAZVAN 52=20240228-17:00:10.471 56=BUX
262=1709139610471 263=0 264=0 146=1 55=BET 267=1 269=3 10=224
```

Request response, MarketDataIncrementalRefresh[X]:

```
8=FIXT.1.1 9=128 35=X 34=1534 49=BUX 52=20240228-17:00:10.836 56=RAZVAN 1128=7 268=1
279=0 269=3 55=BET 270=5892.1 272=20240228 273=08:00:04.005 10=133
```

MDUpdateAction (tag:279) = 0 (NEW)

MDEntryType (tag:269) = 1 (INDEX VALUE)

```
8=FIXT.1.1 9=128 35=X 34=1535 49=BUX 52=20240228-17:00:10.836 56=RAZVAN 1128=7 268=1
279=0 269=3 55=BET 270=5892.1 272=20240228 273=08:00:04.062 10=137
```

MDUpdateAction (tag:279) = 0 (NEW)

MDEntryType (tag:269) = 3 (INDEX VALUE)

MarketDataRequest[V] Snapshot plus updates

Request example:

```
8=FIXT.1.1 9=119 35=V 34=3 49=RAZVAN 52=20240228-17:27:48.340 56=BUX 262=1709141268340
263=1 264=0 265=1 266=Y 146=1 55=BET 267=1 269=3 10=190
```

Response example:

```
8=FIXT.1.1 9=172 35=W 34=41 49=BUX 52=20240228-17:27:48.480 56=RAZVAN 1128=7 55=BET
262=1709141268340 779=20240228-17:27:48.479 1021=1 268=1 269=3 270=5891.96
272=20240228 273=15:36:08.744 10=062
```

Symbol (tag:55) = BET MDEntryType (tag:269) = 3 (INDEX VALUE)

End of response: 8=FIXT.1.1 9=121 35=W 34=42 49=BUX 52=20240228-17:27:48.480 56=RAZVAN 1128=7 55=* 262=1709141268340 779=20240228-17:27:48.480 268=1 269=J 10=090

MarketDataRequest[V] Snapshot for Settle Price[269=6]

Request example:

8=FIXT.1.1 9=105 35=V 34=2 49=RAZVAN 52=20240807-14:01:00.267 56=BUX 262=1723039260252

263=0 264=0 146=1 55=* 267=1 269=6 10=184

Response example:

```
8=FIXT.1.1 9=129 35=W 34=2 49=BUX 52=20240807-14:01:01.078 56=RAZVAN 1128=7 55=ARAX
262=1723039260252 779=20240806-21:00:15.416 268=1 269=6 270=2 10=254
8=FIXT.1.1 9=131 35=W 34=3 49=BUX 52=20240807-14:01:01.078 56=RAZVAN 1128=7 55=VISA
262=1723039260252 779=20240806-21:00:15.361 268=1 269=6 270=710 10=100
8=FIXT.1.1 9=132 35=W 34=4 49=BUX 52=20240807-14:01:01.078 56=RAZVAN 1128=7 55=OMV
262=1723039260252 779=20240806-21:00:15.362 268=1 269=6 270=146.4 10=139
8=FIXT.1.1 9=130 35=W 34=5 49=BUX 52=20240807-14:01:01.078 56=RAZVAN 1128=7 55=DAI
262=1723039260252 779=20240806-21:00:15.362 268=1 269=6 270=323 10=001
8=FIXT.1.1 9=131 35=W 34=6 49=BUX 52=20240807-14:01:01.078 56=RAZVAN 1128=7 55=BAC
262=1723039260252 779=20240806-21:00:15.362 268=1 269=6 270=58.4 10=050
8=FIXT.1.1 9=130 35=W 34=7 49=BUX 52=20240807-14:01:01.078 56=RAZVAN 1128=7 55=SIE
262=1723039260252 779=20240806-21:00:15.362 268=1 269=6 270=500 10=019
8=FIXT.1.1 9=131 35=W 34=8 49=BUX 52=20240807-14:01:01.078 56=RAZVAN 1128=7 55=INL
262=1723039260252 779=20240806-21:00:15.363 268=1 269=6 270=82.6 10=081
8=FIXT.1.1 9=130 35=W 34=9 49=BUX 52=20240807-14:01:01.078 56=RAZVAN 1128=7 55=SAP
262=1723039260252 779=20240806-21:00:15.363 268=1 269=6 270=255 10=032
8=FIXT.1.1 9=131 35=W 34=10 49=BUX 52=20240807-14:01:01.078 56=RAZVAN 1128=7 55=ADS
262=1723039260252 779=20240806-21:00:15.363 268=1 269=6 270=370 10=059
8=FIXT.1.1 9=134 35=W 34=11 49=BUX 52=20240807-14:01:01.078 56=RAZVAN 1128=7 55=HEIA
262=1723039260252 779=20240806-21:00:15.363 268=1 269=6 270=210.1 10=214
8=FIXT.1.1 9=123 35=W 34=2037 49=BUX 52=20240807-14:01:01.165 56=RAZVAN 1128=7 55=*
262=1723039260252 779=20240807-14:01:01.165 268=1 269=J 10=152
```

Note: updates will be of type Market Data Full Snapshot, and the end will be market with Empty book (269 =J)

MarketDataRequest[V] Subscription for Settle Price[269=6]

Request example:

```
8=FIXT.1.1 9=105 35=V 34=5 49=RAZVAN 52=20240807-14:02:23.403 56=BUX 262=1723039343403
263=1 264=0 146=1 55=* 267=1 269=6 10=186
```

Response example:

```
8=FIXT.1.1 9=97 35=X 34=2042 49=BUX 52=20240807-14:03:18.225 56=RAZVAN 1128=7 268=1
279=0 269=6 55=CLUJ 270=1.45 10=149
```

Note: the updates will be of type Market Data Incremental Update when the settle price is set by the Exchange

MarketDataRequest[V] Snapshot for Open interest [269=C]

Request example:

```
8=FIXT.1.1  9=105  35=V  34=2  49=RAZVAN  52=20240802-14:49:35.314  56=BUX
262=1722610175300  263=0  264=0  146=1  55=*  267=1  269=C  10=198
```

Response example:

```
8=FIXT.1.1  9=115  35=W  34=2  49=BUX  52=20240802-11:49:38.799  56=RAZVAN  1128=7
55=BET25JUN  779=20240802-11:11:09.641  268=1  269=C  271=0  10=174
8=FIXT.1.1  9=115  35=W  34=3  49=BUX  52=20240802-11:49:38.800  56=RAZVAN  1128=7
55=BET25MAR  779=20240802-11:11:08.641  268=1  269=C  271=0  10=144
8=FIXT.1.1  9=115  35=W  34=4  49=BUX  52=20240802-11:49:38.800  56=RAZVAN  1128=7
55=BVB14DEC  779=20240801-18:00:16.466  268=1  269=C  271=0  10=130
8=FIXT.1.1  9=115  35=W  34=5  49=BUX  52=20240802-11:49:38.801  56=RAZVAN  1128=7
55=FP24DEC  779=20240802-11:11:05.641  268=1  269=C  271=25  10=109
8=FIXT.1.1  9=116  35=W  34=6  49=BUX  52=20240802-11:49:38.801  56=RAZVAN  1128=7
55=H2O24DEC  779=20240802-11:11:04.643  268=1  269=C  271=41  10=161
8=FIXT.1.1  9=118  35=W  34=7  49=BUX  52=20240802-11:49:38.802  56=RAZVAN  1128=7
55=TLV24SEP1  779=20240802-11:11:01.646  268=1  269=C  271=930  10=086
8=FIXT.1.1 9=121 35=W 34=17 49=BUX 52=20240802-11:49:38.806 56=RAZVAN 1128=7 55=*
262=1722610175300 779=20240802-11:49:38.806 268=1 269=J 10=075
```

Note: updates will be of type Market Data Full Snapshot, and the end will be market with Empty book (269 =J)

MarketDataRequest[V] Subscription for Open interest[269=C]

Request example:

```
8=FIXT.1.1 9=105 35=V 34=3 49=RAZVAN 52=20240802-15:09:28.075 56=BUX 262=1722611368061
263=1 264=0 146=1 55=* 267=1 269=C 10=212
```

Response example:

```
8=FIXT.1.1 9=96 35=X 34=15 49=BUX 52=20240806-15:45:39.150 56=RAZVAN 1128=7 268=1
279=0 269=C 55=BET24DEC 271=0 10=141
```

Note: the updates will be of type Market Data Incremental Update when open interest is set

Bandwidth recommendation

The bandwidth recommendation is:

- minimum 40 kbps for 1-second latency during 50% time
- minimum 125 kbps for 1-second latency during 90% time
- minimum 3 Mbps for 1-second latency during 99% time
- minimum 4 Mbps for 1-second latency during 99.99% time

The bandwidth recommendation is specified per only one client connection. If there are two or more connections to market-data fix server, bandwidth should be proportionally increased.