FIGURE

TECHNICAL SPECIFICATIONS – FIX 5.0 INTERFACE

Order Entry & Market Data

Version: 1.34
Date: February 17, 2026

Table of Contents

Change History Log

Version	Date	Description
1.0	2024-07-09	Initial document version
1.1	2024-07-22	Clarified language around Cancel on Disconnect, which currently works for DAY orders only
1.2	2024-09-11	Added tags & enums to spec that are inline with data dictionary
1.22	2024-10-10	Typo fixes
1.23	2024-10-11	Added EVENT SecurityType. Other typo fixes.
1.24	2024-12-13	Added *SecuritySubType (762)*, *LegSecuritySubType (764)* and *TradingSessionID (336)*.
1.25	2025-01-30	• Added *ManualOrderIndicator (1028), CorporateAction (292)*, and MDEntryType = g (Trading Reference Price) • Clarified that *TimeInForce (59)* on an order can be modified.
1.26	2025-02-27	• Removed SUSPEND (S) as a valid *ExecInst (18)* • Added a description of the ClOrdID uniqueness behavior to the “Order Entry and Matching” section
1.27	2025-03-06	Updated description of *BusinessRejectRefID (379)* to explain that it now contains the ClOrdId or MDReqID of the rejected message
1.28	2025-05-05	Updated to reflect *TradingSessionID (336)* type and values
1.29	2025-07-24	Added “Trade Busts” section
1.30	2025-08-07	Aligned TIF GTC/GTD language with online docs
1.31	2025-08-07	Updated Table 4 (Logon Message) to include fields for *Username (553)* and *Password (554)*
1.32	2025-11-25	Clarified language around *MDUpdateAction (279)* = 2 (Delete)
1.33	2025-12-15	Added *ClOrdLinkID (583)* row to the NewOrderSingle [D] table
1.34	2026-02-11	• Updated Order Cancellation section to include Mass Cancel functionality

	• Updated comment on *TradingSessionID (336)* in MarketDataIncrementalRefresh [X] table to reflect that TradingSessionID is provided for all entries other than MDEntryType (269) = 0 (Bid) or 1 (Offer) • Updated NewOrderSingle [D], OrderCancelRequest [F], OrderCancelReplaceRequest [G], MarketDataRequest [V], and SecurityListRequest [x] tables to reflect that either *Symbol (55)* or *SecurityID (48)* + *SecurityIDSource (22)* are required • Added "3 = Done For Day" as an *ExecType (150)* to the ExecutionReport [8] table • Added "8 = Market / exchange option" as an *ExecRestatementReason (378)* to the ExecutionReport [8] table • Added "25 = Insufficient credit limit" as an *OrdRejReason (103)* to the ExecutionReport [8] table and as a *CxlRejReason (102)* to the OrderMassCancelRequest [q] table, and removed unused OrdRejReason and CxlRejReason enums

Support Contacts

For general inquiries, help with Figure's APIs, or any other support-related matters, please reach out to:

Trade Enquiries: support@figuremarkets.com

Production Support: support@figuremarkets.com

API Developer Support: apisupport@figuremarkets.com

Introduction

The purpose of this document is to outline the trading functionality available on Figure Markets via a FIX trading protocol.

Platform Overview

Figure Markets leverages MPC technology to enable self-custody. In addition to self-custodied Figure wallets, for each supported L1 network, Figure Markets creates dedicated MPC custodial accounts. The MPC custodial accounts allow for the deposit and withdrawal of L1 assets. Additionally, we employ decentralization in the MPC custodial accounts to ensure that no single entity can move assets without customer consent.

The exchange supports high-frequency trading via matching off-chain and transparency of ownership through on-chain settlement. It allows retail and institutional investors to trade assets from Layer 1 (L1) blockchains like Bitcoin, Ethereum, and Provenance.

To facilitate high frequency matching alongside on-chain settlement, tokens held in Figure Markets accounts must be committed to the exchange. This allows the blockchain to settle matched trades on chain, periodically, while preventing double-spending. To withdraw assets from your account, first uncommit them from your Figure Markets account.

The Figure Markets exchange operates 24 hours a day, with a scheduled maintenance window between 4:45 - 5:00 PM Central Time each day.

Platform Network Connectivity

FIX connections are for institutions and to get started with connectivity a firm must be approved. The application can be found here: https://www.figuremarkets.com/universal-passport/auth0/login.

The Figure Markets onboarding team can assist with the application process, setting up your environment, and whitelisting your IP Address.

Settlement

Settlement of trades occurs on the Provenance Blockchain periodically throughout the day. Under normal trading circumstances, you can expect trades to settle minutes after a match is found.

Firm, User and Account Identifiers

The FIGURE Platform exposes two FIX gateways to participants; an order management gateway, and a second gateway to receive market data. Additional drop-copy sessions can be provided upon request.

Note that should the Exchange offer direct market access (DMA) to their underlying customer, then each DMA customer should have their own dedicated pair of FIX gateways.

An example FIX session configuration is shown below. Note the network details and Sender/TargetCompID (in red) will be provided by the exchange operator.

Snippet: Example QuickFIX session configuration

```
[DEFAULT]
  ConnectionType=initiator
  SocketConnectHost=12.12.12.12
  SocketConnectPort=13001
  BeginString=FIXT.1.1
  DefaultApplVerID=9
  SenderCompID=SENDERCOMP1
[SESSION]
  TargetCompID=EXCHANGECOMPID
```

The FIGURE platform also validates individual users (traders) using *SenderSubID (50)*, and customer accounts using *Account (1)* which are validated on order entry. **Please contact FIgure Markets to allocate these codes.**

Symbology

FIGURE uses a simple string identifier to instruments trading on the platform, which is required to identify instruments in the API using *Symbol (55)* or *SecurityID (48)*. There is currently no support to identify instruments using any other common identifiers such as CUSIP, ISIN or Bloomberg code.

A list of instruments on the platform can be retrieved using the SecurityListRequest [x] message.

Instrument States

Instrument may the the following states:

Table 1: Instrument States

State	Definition
PENDING	Initial state for a newly created instrument which has not yet begun trading.
OPEN	In this state, the instrument is open for continuous order entry and matching.
CLOSED	In this state, orders can not be entered, modified, or canceled, and no matching occurs. Any existing Day orders will be expired.
PREOPEN	Orders can be entered and modified, but no matching occurs. When the instrument transitions to an OPEN state, the orders entered during PREOPEN will match at a single opening price that is automatically determined by an algorithm that is designed to maximize the volume traded at the open.

MATCH_AND_CLOSE_ AUCTION	This state is similar to PREOPEN, with the exception that matching will occur upon the transition of this state to any other state. This state is useful if you want matching to occur at the end of the state, but you don't want the instrument to be open after.
SUSPENDED	Orders can be canceled but no matching occurs, and no order entry or modification is allowed.
HALTED	This state is similar to SUSPENDED, with the exception that orders cannot be canceled.
EXPIRED	An Instrument moves to this state when its Expiration Date/Time is reached. In this state, any resting orders are expired and no new orders can be entered.
TERMINATED	When an Instrument's Termination Date is reached, the order book is removed from the matching engine, orders are canceled, and positions are closed. Historical data will still remain in FIGURE ledgers.

FIX Notation

Please note the following presentation notes which apply to message definitions and FIX examples throughout this document.

- FIX tag/value pairs are delimited within a TCP connection using the SOH (ascii character 1) character. Since this is a non-printable character, in this document we use the | character instead, and pad each pair with spaces to make them easier to use.
- Components are blocks of FIX tags which appear frequently in the specification (e.g. header and footers which appear on every FIX message). They are defined centrally for convenience and then referenced throughout the document using <> notation.
- Repeating groups of FIX tags appear in various messages. The depth of a repeating group is indicated using the → marker in FIX message definitions.
- References to individual FIX fields (or “tags”) are presented in italic font, with the tag number following the tag name. For example *HeartBtInt (108)*.

FIX Component Definitions

Standard Header & Trailer

Each FIX message sent to and received from the FIGURE platform must start and end with a message header and trailer components.

Table 2: Standard Header component

Tag	Field Name	Req'd	Data Type	Comments
8	BeginString	Y	String	FIXT.1.1
9	BodyLength	Y	Length	Standard FIX message body length
35	MsgType	Y	String	The message type. See relevant section.
49	SenderCompID	Y	String	Sender identity as agreed with the exchange operator.
56	TargetCompID	Y	String	Intended target identity as agreed with the exchange operator.
50	SenderSubID	C	String	Sender sub-identifier representing an individual user (as previously agreed with the exchange operator). **Required on all application messages related to the entry or management of orders.**
57	TargetSubID	N	String	Target sub-identifier representing an individual user (as previously agreed with the exchange operator)
34	MsgSeqNum	Y	SeqNum	FIX Message sequence number
43	PossDupFlag	N	Boolean	Always required for retransmitted messages as the result of a resend request
52	SendingTime	Y	UTCTime	Sending time **in UTC**

Table 3: Standard Trailer component

Tag	Field Name	Req'd	Data Type	Comments
10	Checksum	Y	String	Standard FIX checksum

FIX Session Level Management

Establishing and Maintaining a FIX Session

The Logon [A] message initiates a connection to the FIGURE platform. FIX sessions are typically initiated by customers (as opposed to the exchange platform) and can be made at any time.

Important: Logon [A] messages must NOT specify a SenderSubID [50] in the header. Sending this tag will cause the logon attempt to be rejected.

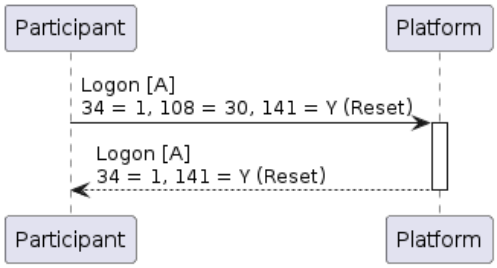
Table 4: Logon [A] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = A
98	EncryptMethod	Y	int	Encrypted messages are not supported 0 = None
108	HeartBtInt	Y	int	A 30-second interval is recommended
141	ResetSeqNumFlag	Y	Boolean	Indicates both sides of a FIX session should reset sequence numbers back to 1 during a normal end of session (and not due to an unintended disconnect). **Recommended**
553	Username	N	String	If the FIX session setting *RequiresAuth* is set, this field will be authenticated against an FIGURE username
554	Password	N	String	If the FIX session setting *RequiresAuth* is set, this field will be authenticated against an FIGURE password
< Standard Trailer >		Y		

Example 1: Logon message

```
8=FIXT.1.1 | 9=76 | 35=A | 49=SENDER | 56=TARGET | 34=1 | 52=20240516-14:14:53 |
98=0 | 108=30 | 1137=9 | 141=Y | 10=132 |
```

Figure 1: Successful Logon



Under normal circumstances, both FIX engines on either side of the connection will regularly exchange Heartbeat messages. The frequency of such a message exchange is determined by the *HeartBtInt (108)* value indicated in the Logon [A] message.

Regular application messages qualify as a heartbeat, i.e both sides of the connection are behaving nominally if a message is received during the *HeartBtInt (108)*. If no application messages are received during the interval, the Heartbeat [0] message is used as described below.

Table 5: Heartbeat [0] message

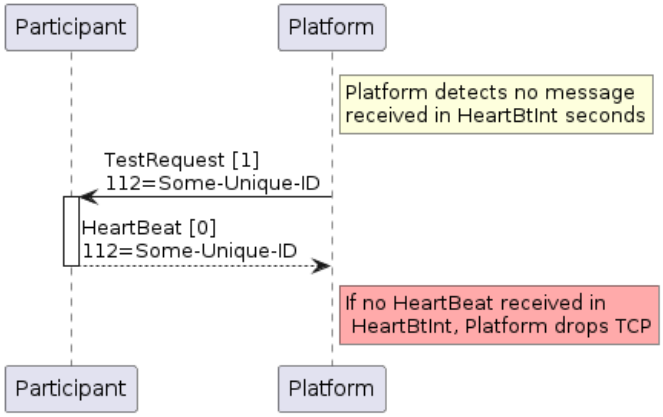
Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 0 (zero)
112	TestReqID	N	String	Required when the heartbeat is the result of a TestRequest [1] message
< Standard Trailer >		Y		

At any point either counterparty can force the opposing FIX engine to send a Heartbeat [0] message by submitting a TestRequest [1] message containing a *TestReqID (112)* value, which should be echoed in the Heartbeat [0] response.

Table 6: TestRequest [1] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 1
112	TestReqID	Y	String	ID to be returned in the resulting Heartbeat [0] response
< Standard Trailer >		Y		

Figure 2: Using TestRequest [1] to request Heartbeat [0] from other side



Cancel on Disconnect

Cancel on Disconnect is an optional feature which can act as an automatic risk control for participants in the event that application connectivity is lost. Examples may include where a Participant's application stops responding to messages for a certain period. In these circumstances, the FIGURE platform will automatically cancel all open (unexecuted) DAY orders for the Participant while GTC and GTD orders continue to rest.

Note that a network-level disconnect instantly triggers this Cancel on Disconnect functionality. After reconnecting, missed messages will be replayed including execution reports for any canceled orders.

Cancel on Disconnect is disabled by default and enabled on a session-by-session basis. Contact your exchange operator representative to enable this functionality.

Sequence Number Tracking, Recovery, and Reset

The FIX protocol uses simple, incrementing *MsgSeqNum (34)* values (carried in the StandardHeader of each message) to both detect and request retransmission of missed messages.

Each FIX engine will maintain a simple message count of outbound and inbound messages, values which should increment by one for each message sent or received per FIX session. Should a message be received that does not match the **expected** sequence number given, then it is possible that one or more messages were lost.

In that situation, a ResendRequest [2] message may be sent to request retransmission of a specified range of messages identified by their *MsgSeqNum (34)* value.

Table 7: ResendRequest [2] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 2
7	BeginSeqNo	Y	SeqNum	*MsgSeqNum (34)* of first message in the range to be resent (inclusive)
16	EndSeqNo	Y	SeqNum	*MsgSeqNum (34)* of the last message in the range to be resent (inclusive), or "0" to request resend all messages after the indicated *BeginSeqNo (7)*
< Standard Trailer >		Y		

The expected response to a ResendRequest [2] message is a stream of application messages with *PossDupFlag (43)* set to "Y" (yes) in the header to indicate that this is a potentially duplicative message.

It may be appropriate in some situations for the opposite FIX engine to refuse to replay the message. This is typically the case for administrative messages which either have limited value (e.g., Heartbeat messages) or might cause confusion (e.g., Logon messages). In such circumstances, the opposite FIX engine may respond with a SequenceReset [4] message (below) with *GapFillFlag (123)* set to "Y".

The SequenceReset [4] message can be used in two distinct situations:

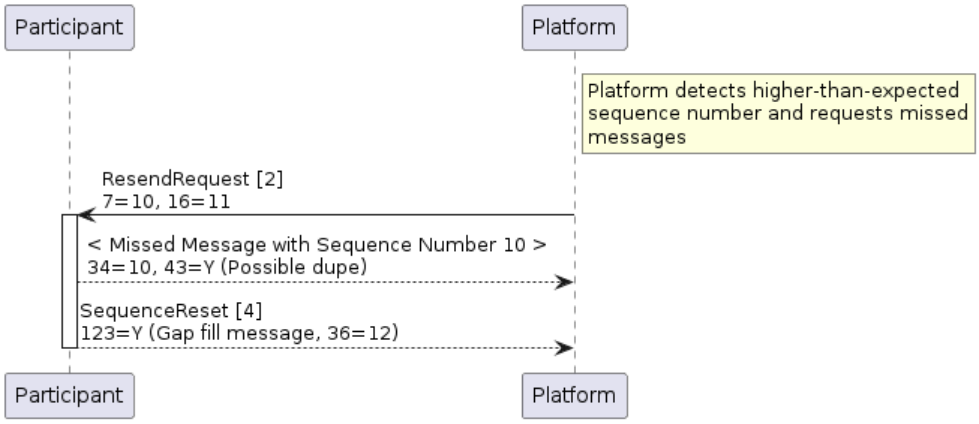
1. To request that the counterparty adjusts their internal sequence number to the indicated *NewSeqNo (36)* value. In this case, *GapFillFlag (123)* is either not present or set to "N" (no).

2. In the message replay situation described above, to replace a FIX message which will not be re-sent. In this case, *GapFillFlag (123)* should be set to "Y" and the *NewSeqNo (36)* field will contain the next sequence number to be sent.

Table 8: SequenceReset [4] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 4
123	GapFillFlag	N	Boolean	Indicates that the Sequence Reset message is replacing administrative or application messages which will not be resent.
36	NewSeqNo	Y	SeqNum	New sequence number
< Standard Trailer >		Y		

Figure 3: ResendRequest [2] message triggers re-transmission of missed messages



Terminating a FIX Session

The Logout [5] message initiates or confirms the termination of a FIX session. Customers may log out of their FIX sessions at any time.

Table 9: Logout [5] message

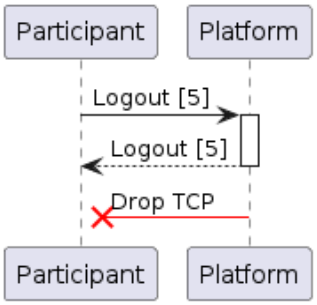
Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 5
58	Text	N	String	Free format text string
< Standard Trailer >		Y		

The expected response to a successful Logout [5] message is a reciprocal Logout [5] message, after which the TCP connection should be dropped.

Example 2: Logout message

```
8=FIXT.1.1 | 9=51 | 35=5 | 34=2 | 49=SENDER | 52=20240516-14:17:38 | 56=TARGET |
10=237 |
```

Figure 4: Graceful Logout sequence



Participants are recommended to schedule a graceful logout before the start of a scheduled maintenance window. If this is not received, then the FIGURE Platform will initiate the logout process.

IMPORTANT: If Cancel on Disconnect is enabled, any open (unexecuted) DAY orders in the FIGURE platform are automatically canceled when a FIX session terminates for any reason, including a graceful logout. GTC and GTD orders will continue to rest.

General Error Handling

Most transactional messages in FIX are rejected using an application-level message (as described in various sections which follow).

In the event that one of these messages cannot be used to reject a message, however, then the FIGURE Platform may return either a Reject [3] message, or a BusinessMessageReject [j] message as described below.

- Reject [3] messages are commonly used to reject messages where the message type is known, but there is something syntactically wrong with its content (e.g. incorrect repeating group).
- BusinessMessageReject [j] is a more generic message used to reject errors such as unsupported message type or malformed messages.

Table 10: Reject [3] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 3
45	RefSeqNum	Y	SeqNum	The message sequence number being rejected
371	RefTagID	N	int	The tag (field) number being rejected
373	SessionRejectReason	N	int	Reason for the rejection. 0 = Invalid tag number 1 = Required tag missing

Tag	Field Name	Req'd	Data Type	Comments
				2 = Tag not defined for this message type 3 = Undefined tag 4 = Tag specified without value 5 = Value is incorrect 6 = Incorrect data format for value 9 = CompID problem 10 = SendingTime accuracy problem 11 = Invalid MsgType 13 = Tag appears more than once 14 = Tag specified out of required order 15 = Repeating group fields out of order 16 = Incorrect NumInGroup count 99 = Other
58	Text	N	String	Optional string to further describe the error
< Standard Trailer >		Y		

Example 3: Reject [3] example due to bad limit price

```
8=FIXT.1.1 | 9=127 | 35=3 | 34=39 | 49=TARGET | 52=20240517-19:08:47 | 56=SENDER | 45=42 | 58=Value is incorrect (out of range) for this tag | 371=44 | 372=D | 373=5 | 10=237 |
```

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = j
45	RefSeqNum	N	SeqNum	The message sequence number being rejected
58	Text	N	String	String to further describe the error
372	RefMsgType	Y	String	The MsgType being rejected
379	BusinessRejectRefID	N	String	When a FIX gateway rejects a message with a BusinessMessageReject, it provides tag 379 (`BusinessRejectRefID`) on the BusinessMessageReject and populates it with the ClOrdID or MDReqID to allow FIX clients to quickly determine which message was rejected.
380	BusinessRejectReason	Y	int	The reason for the rejection. 0 = Other 1 = Unknown ID 2 = Unknown Security 3 = Unsupported Message Type 4 = Application Not Available (downtime) 5 = Conditionally required field missing 6 = Not authorized 18 = Invalid price increment (tick size)
< Standard Trailer >		Y		

Table 11: BusinessMessageReject [j] message

Example 4: BusinessMessageReject [j] example due to bad role permissions

```
8=FIXT.1.1 | 9=116 | 35=j | 34=5 | 49=TARGET | 52=20240516-14:19:40 | 56=SENDER |
45=6 | 58=Supervising firms cannot perform this action | 372=x | 380=6 | 10=082 |
```

Order Entry and Matching

Overview

FIGURE operates as a continuous central limit order book to match orders within the system according to price-time priority:

- Higher-priced orders to buy have priority over lower-priced bids,
- Lower-priced offers to sell buy have priority over higher-priced sells,
- Within a price level, older orders have priority over newer orders,

- Should an existing order increase its quantity (at the same price), it is assigned a new timestamp and therefore loses time priority, but quantity decreases retain time priority.

ClOrdID (Client Order ID)

ClOrdID is a unique, participant-created identifier for this Order. Uniqueness must be guaranteed across a session (i.e. between logon and logout), which may span multiple days.

FIGURE caches ClOrdIDs of active orders to ensure that you can cancel-replace orders by specifying the OrigClOrdId. As soon as a GTC order becomes inactive, its ClOrdID can be re-used.

Self-Match Prevention

The FIGURE platform offers self-match prevention logic which can be enabled either at the FIX session level (automatically applied to all orders entered through a session), or on a per-order basis using the *SelfMatchPreventionID (7928)* and *SelfMatchPreventionInstruction (8000)* tags.

This instruction will automatically cancel one or both orders (without execution) which would potentially be involved in a self-match. The options are:

1. The aggressor (new) order will be canceled.
2. The passive (existing) order is canceled.

See Self-Match Prevention for further details and examples of this behavior. Contact the exchange operator to enable the setting at the FIX session level.

Tick Sizes / Minimum Quantity Increments

Each instrument on the FIGURE Platform has a minimum price increment (“tick size”) and minimum quantity increment (“lot size”). These instrument-level settings are indicated in the SecurityList [y] message in the *MinPriceIncrement (969)* and *MinTradeVol (562)* fields respectively.

Order with a price and/or quantity which do not comply with these increments will be rejected using BusinessMessageReject [j] with reason 13 = Incorrect quantity or 18 = Invalid price increment (tick size).

Order Types, Time In Force & Execution Instructions

The intended behavior of orders entering the FIGURE Platform is controlled using a combination of *OrderType (40)*, *TimeInForce (59)* and *ExecInst (18)* fields. These are explained in turn below.

Table 12: OrdType [40] definitions

Order Type	Value	Definition
LIMIT	2	A priced order that can only trade at a price equal to or better than the price specified.
MARKET TO LIMIT	K	An order that fills as far as possible at the best price(s) in the market. Should the order volume exceed that available in the order book, then the remaining order quantity is converted into a Limit order with the price equal to the last fill price (subject to Time In Force instructions). In the case where there is no market, the order will be rejected with reason "No liquidity for market order".
STOP	3	An order which initially enters the system as hidden, but which activates (converts to active) once the price indicated by *StopPx (99)* is triggered in the market. At this point, the order is converted into a Market to Limit order with the indicated quantity.
STOP LIMIT	4	An order similar to STOP, but where the converted order is a Limit order with the indicated *Price (44)*.

Looking for a “classic” Market Order? Simply enter a Market-to-Limit order with IOC Time in Force condition.

Table 13: TimeInForce [59] conditions

Time In Force	Value	Definition
DAY	0	The order remains open for the current trading day only.
GOOD TILL CANCEL (GTC)	1	The order remains available for execution until fully executed or canceled.
IMMEDIATE OR CANCEL (IOC)	3	The order is immediately executed as far as possible upon entry, with all unfilled quantity expired. Can be used in conjunction with *MinQty (110)* to specify a minimum quantity which must be executed immediately.
FILL OR KILL (FOK)	4	If the entire order quantity can not be satisfied immediately, then the order is canceled in full.
GOOD TILL DATE (GTD)	6	The trade is active until a specific date and time (expressed in UTC) as indicated in *ExpireTime (126)*.

In addition to order-level Time In force conditions, participants can also specify a minimum order quantity using *MinQty (110)*, which requires that an order receives at least a minimum execution quantity upon entry (possibly in multiple fills). When used in combination with Time In Force, this field offers fine-grained control over the conditions under which an order can participate. For example:

- An order marked as IOC with *MinQty (110)* of 200 must receive an immediate, minimum trade quantity of 200. If this is not possible, then the entire order immediately expires.

- In the case of orders marked with a FOK *TimeInForce (59)*, *MinQty (110)* can be included but has no material effect, as the FOK requires a full fill upon entry.
- A Good Till Cancel order with a *MinQty (110)* specified must trade at least the specified quantity upon first entry into the order book, otherwise the order expires.

Note that *MinQty (110)* only limits order behavior as it enters the order book; it does not prevent smaller trades occurring against the order once it is resting on the order book.

Table 14: ExecInst [18] definitions

Exec Instruction	Value	Definition
ALL OR NONE	G	Require that either (a) all of the order quantity is filled immediately, or (b) none of it should trade even partially. Marking an order with this ExecInst instruction is functionally-equivalent to setting Time In Force to FOK.
IGNORE PRICE VALIDITY CHECKS	c	Exempt this order from absolute price limits, relative price limits, order size limits, and total notional limits. Only allowed for market-to-limit orders to sell. This flag exists to support quickly liquidating a position.
PARTICIPATE DON'T INITIATE	6	Require that the order is only accepted if it would NOT immediately match. This guarantees that the order will always be the passive side in any trades. An order that would result in a match would be rejected with reason "Order may participate but not initiate in the market". Should not be used in combination with *MinQty (110)* or conditions which require immediate executions - *ExecInst (18)* = G (All or None), or *TimeInForce (59)* = 3 or 4 (IOC or FOK).
SINGLE_EXECUTION_REQUESTED_FOR_BLOCK_TRADE	j	A flag for submitting block trades to FIGURE.
BEST_LIMIT	R	A flag that if set indicates that the price of a limit order shall be set to the price at the top of the book on the same side as this order.
IMMEDIATELY_EXECUTABLE_LIMIT	T	A flag that if set indicates that the price of a limit order shall be set to the price at the top of the book on the opposing side as this order, thus able to immediately match.

Order Entry

Participants may place orders into the FIGURE platform order book to buy or sell securities using a NewOrderSingle [D] message.

Table 15: NewOrderSingle [D] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = D
11	ClOrdID	Y	String	Unique, participant-created identifier for this Order. Uniqueness must be guaranteed across a session (i.e. between logon and logout), which may span multiple days.
1	Account	N	String	Account reference as previously advised to the exchange operator
18	ExecInst	N	MultipleChar	Instructions for order handling. Note that Price Validity Checks can only be ignored (c) for market-to-limit orders to sell. G = All or None b = Strict Limit c = Ignore Price Validity Checks j = Single Execution Requested For Block Trade 6 = Participate Don't Initiate R = Best Limit S = Suspend T = Immediately Executable Limit
22	SecurityIDSource	C	String	Required if *SecurityID (48)* is provided. The only acceptable value is 8 (Exchange Symbol), even if *security_id_source* on the FIGURE instrument is set to something other than Exchange Symbol.
110	MinQty	N	Qty	Minimum order quantity that must be executed upon entry (or else the whole order is immediately canceled).
48	SecurityID	C	String	Instrument symbol. Required if *SecurityIDSource (22)* is provided. Either *SecurityID (48)* or *Symbol (55)* must be provided. Even if *security_id* on the FIGURE instrument is defined as something other than the FIGURE instrument symbol, this still must be set to the instrument symbol.
55	Symbol	C	String	Instrument symbol. Either *SecurityID (48)* or *Symbol (55)* must be provided.
54	Side	Y	char	1 = Buy 2 = Sell
60	TransactTime	N	UTCTime	Timestamp of order entry in UTC.
38	OrderQty	Y	Qty	Order quantity. Can be a decimal.

Tag	Field Name	Req'd	Data Type	Comments
40	OrdType	Y	Char	Order type. 2 = Limit 3 = Stop 4 = Stop limit K = Market with left-over as limit
44	Price	N	Price	Price per share/unit. Required where *OrdType (40)* = 2 (Limit) or 4 (Stop Limit).
99	StopPx	N	Price	Stop price at which to trigger the stop order. Required for *OrdType (40)* = 3 (Stop) or 4 (Stop Limit). Must be greater than or equal to *Price (44)* for buy order, or less than or equal to *Price (44)* for sell orders.
167	SecurityType	N	String	Type of instrument CS = Common Stock EVENT = Event contract FUT = Future FXSPOT = FX Spot (incl. crypto) FXSWAP = FX Swap MLEG = Multileg Instrument OOF = Options on Futures OPT = Option NONE = Other
762	SecuritySubType	N	String	Sub-type qualification/identification of the SecurityType <167> (e.g. for SecurityType="REPO"), or the CFICode <461> if SecurityType is not specified. If specified, SecurityType or CFICode is required.
581	AccountType	N	Int	1=CUSTOMER 2=NON_CUSTOMER 3=HOUSE_TRADER 4=FLOOR_TRADER 6=NON_CUSTOMER_CROSS_MARGINED 7=HOUSE_TRADER_CROSS_MARGINED 8=JOINT_BACK_OFFICE 9=EQUITIES_SPECIALIST 10=OPTIONS_MARKET_MAKER 11=OPTIONS_FIRM_ACCOUNT 12=AGGREGATED_CUSTOMER_AND_NON_CUSTOMER 13=AGGREGATED_MULTIPLE_CUSTOMERS 14=LIQUIDITY_PROVIDER 15=OPERATING 16=CLEARING_FUND 17=FUTURES_MARKET_MAKER
582	CustOrderCapacity	N	Int	1=OWN_ACCOUNT 2=PROPRIETARY_ACCOUNT 3=FINANCIAL_ADVISOR 4=ALL_OTHER 5=RETAIL_CUSTOMER

Tag	Field Name	Req'd	Data Type	Comments
583	ClOrdLinkID	N	String	Permits order originators to tie together groups of orders in which trades resulting from orders are associated for a specific purpose, for example the calculation of average execution price for a customer or to associate lists submitted to a broker as waves of a larger program trade.
453	NoPartyIDs	N	NumingGroup	Number of PartyID (448), PartyIDSource (447), and PartyRole (452) entries
→ 448	PartyID	N	String	Party identifier/code
→ 447	PartyIDSource	N	char	D=Proprietary
→ 452	PartyRole	N	Int	1=EXECUTING_FIRM 3=CLIENT_ID 24=CUSTOMER_ACCOUNT
59	TimeInForce	N	char	0 = Good for day [Default] 1 = Good till cancel 3 = Immediate or cancel 4 = Fill or kill 6 = Good till date
126	ExpireTime	N	UTCTime	Order expiry date and time for orders where TimeInForce = Good Till Date
1028	ManualOrderIndicator	N	Boolean	Indicates if the order was initially received manually (as opposed to electronically)
6127	ConditionTriggerMethod	N	int	The reference price used to trigger the stop order. Applicable to Stop orders only. 2 = Last price [Default] 5 = Settlement price Settlement price may be the preferred trigger method for markets where settlement price is updated frequently from a price oracle.
7928	SelfMatchPreventionID	N	String	Unique identifier to be returned in the case of a self-match prevention cancellation. The same ID must be present on all orders where self-match prevention is desired.
8000	SelfMatchPreventionInstruction	C	String	Self-match instruction. O = Cancel oldest (resting) order N = Cancel newest (aggressive) order
< Standard Trailer >		Y		

Example 5: Entry of a new limit order to buy 1,000 shares at 50.00

```
8=FIXT.1.1 | 9=123 | 35=D | 49=SENDER | 56=TARGET | 34=16 | 50=SENDERSUB |
52=20240517-19:00:28 | 11=1182560819 | 21=1 | 55=GOOG | 54=1 | 40=2 | 44=50 |
38=1000 | 1=ACCT | 10=166 |
```

Order Acknowledgement & Execution

The ExecutionReport [8] message is used to acknowledge various order lifecycle events including the acceptance, rejection, and expiry of orders, as well as providing details of matches (fills) against orders.

All orders entering the Platform are first either accepted or rejected using an Execution Report [8]. The order may then receive further Execution Reports [8] as necessary given their marketability and *TimeInForce (59)* constraints. For example:

1. A badly-formatted order which fails validation upon entry will receive an ExecutionReport [8] with *OrdStatus (39)* = 8 (Rejected).
2. A good for day order which is not immediately executable will receive an ExecutionReport [8] with *OrdStatus (39)* = 0 (New). Should it receive executions during the course of the day, then these will be notified in subsequent Execution Report [8] messages with *OrdStatus (39)* = 1 (Partially Filled) and/or 2 (Fully Filled).
3. An order with *TimeInForce (59)* = 6 (good till date) will receive an ExecutionReport [8] with *OrdStatus (39)* = 0 (New) upon entry. If it remains unexecuted, it will expire at the indicated *ExpireTime (126)* with an Execution Report [8] message with *OrdStatus (39)* = C (Expired).
4. A Fill Or Kill order with *TimeInForce (59)* = 4 which can not be fully executed will first receive an ExecutionReport [8] with *OrdStatus (39)* = 0 (New) and then an immediate Execution Report [8] message with *OrdStatus (39)* = C (Expired).
5. An order with *TimeInForce (59)* = 3 (immediate or cancel) which can be partially filled upon entry will receive at least three ExecutionReport [8] messages; the first will indicate *OrdStatus (39)* = 0 (New), the next messages will detail the fill(s), and the final Execution Report [8] message will expire the remainder with *OrdStatus (39)* = C (Expired).

Table 16: ExecutionReport [8] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 8
1	Account	N	String	Account reference if indicated on the original order
6	AvgPx	Y	Price	Volume-weighted average price of all trades against this order. May be zero for unexecuted orders.

Tag	Field Name	Req'd	Data Type	Comments
11	ClOrdID	Y	String	The participant-assigned ClOrdID value as sent on the last order action message from the Participant (new order, amendment or cancel).
14	CumQty	Y	Qty	Cumulative quantity so far for this order. May be zero for unexecuted orders.
17	ExecID	Y	String	Unique identifier for the Execution Report as assigned by FIGURE. Typically a 13-character alphanumeric string.
22	SecurityIDSource	Y	String	The *security_id_source* on the FIGURE instrument, if defined. If undefined, this is set to 8 (Exchange Symbol). 1 = CUSIP 2 = SEDOL 3 = QUIK 4 = ISIN 5 = RIC 6 = ISO Currency Code 7 = ISO Country Code 8 = Exchange Symbol 9 = Consolidated Tape Association A = Bloomberg Symbol B = Wertpapier C = Dutch D = Valoren E = Sicovam F = Belgian G = Common H = Clearing House I = ISDA FpML Product Specification J = Option Price Reporting Authority K = ISDA FpML Product URL L = Letter of Credit M = Marketplace Assigned Identifier N = Markit RED Entity CLIP P = Markit RED Pair CLIP Q = CFTC Commodity Code R = ISDA Commodity Reference Price S = Financial Instrument Global Identifier (FIGI) T = Legal Entity Identifier (LEI) U = Synthetic V = Fidessa Instrument Mnemonic W = Index Name X = Uniform Symbol Y = Digital Token Identifier
31	LastPx	Y	Price	Price of this last fill. Will be zero for messages not relating to a trade.
32	LastQty	Y	Qty	Quantity traded on this last fill. Will be zero for messages not relating to a trade.

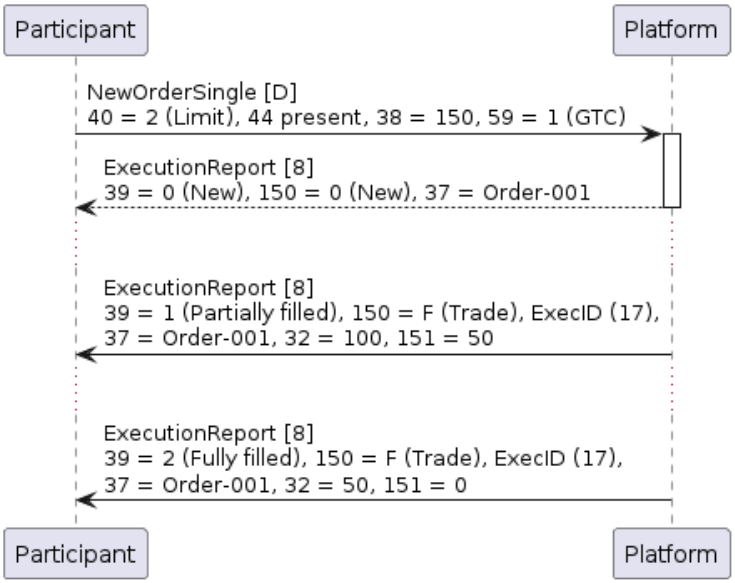
Tag	Field Name	Req'd	Data Type	Comments
37	OrderID	Y	String	Unique identifier for Order as assigned by FIGURE. Typically a 13-character alphanumeric string.
38	OrderQty	Y	Qty	Total order quantity (amended as necessary)
39	OrdStatus	Y	char	The latest status of the order after any changes have been applied. 0 = New 1 = Partially filled 2 = Fully filled 3 = Done For Day 4 = Canceled 6 = Pending Cancel 8 = Rejected A = Pending New C = Expired E = Pending Replace
40	OrdType	Y	char	The type of order. 2 = Limit 3 = Stop 4 = Stop Limit K = Market with left over as limit
41	OrigClOrdID	N	String	Sent in the case of order amendment or cancellation. References the prior *ClOrdID (11)* value that the action amended/canceled.
44	Price	Y	Price	Order limit price (amended as necessary)
48	SecurityID	Y	String	The *security_id* on the FIGURE instrument, if defined. If undefined this will match *Symbol (55).*
54	Side	Y	char	1 = Buy 2 = Sell
55	Symbol	Y	String	Instrument symbol
167	SecurityType	N	String	Type of instrument CS = Common Stock EVENT = Event contract FUT = Future FXSPOT = FX Spot (incl. crypto) FXSWAP = FX Swap MLEG = Multileg Instrument OOF = Options on Futures OPT = Option NONE = Other
762	SecuritySubType	N	String	Sub-type qualification/identification of the SecurityType <167> (e.g. for SecurityType="REPO"), or the CFICode <461> if SecurityType is not specified. If

Tag	Field Name	Req'd	Data Type	Comments
				specified, SecurityType or CFICode is required.
460	Product	N	Int	Indicates the type of product the security is associated with. 1=AGENCY 2=COMMODITY 3=CORPORATE 4=CURRENCY 5=EQUITY 6=GOVERNMENT 7=INDEX 8=LOAN 9=MONEYMARKET 10=MORTGAGE 11=MUNICIPAL 12=OTHER 13=FINANCING 14=ENERGY
59	TimeInForce	Y	char	Echoed from New Order Single 0 = Good for day 1 = Good till cancel 3 = Immediate or cancel 4 = Fill or kill 6 = Good till date
60	TransactTime	Y	UTCTime	Timestamp when the business transaction represented by the message occurred.
99	StopPx	Y	Price	Order stop price (amended as necessary). Will be zero for non stop orders.
103	OrdRejReason	N	int	Rejection reason (where OrdStatus = Rejected) 0 = Broker/Exchange option 2 = Exchange closed 25 = Insufficient credit limit
119	SettlCurrAmt	N	Amt	Present on trades only. Total amount of this last fill. Equal to *LastPx (31)* x *LastQty (32)*
126	ExpireTime	N	UTCTime	Order expiry date (amended as necessary).
150	ExecType	Y	char	The reason that the FIGURE Platform sent this Execution Report. 0 = New 3 = Done For Day 4 = Canceled 5 = Replaced 8 = Rejected C = Expired F = Trade

Tag	Field Name	Req'd	Data Type	Comments
151	LeavesQty	Y	Qty	Remaining, unexecuted quantity left on the order. May be zero for fully filled orders.
381	GrossTradeAmt	N	Amt	Present on orders which have been filled. Total amount traded across all fills for this order. Equal to *AvgPx (6)* x *CumQty (38)*.
581	AccountType	N	Int	1=CUSTOMER 2=NON_CUSTOMER 3=HOUSE_TRADER 4=FLOOR_TRADER 6=NON_CUSTOMER_CROSS_MARGINED 7=HOUSE_TRADER_CROSS_MARGINED 8=JOINT_BACK_OFFICE 9=EQUITIES_SPECIALIST 10=OPTIONS_MARKET_MAKER 11=OPTIONS_FIRM_ACCOUNT 12=AGGREGATED_CUSTOMER_AND_NON_CUSTOMER 13=AGGREGATED_MULTIPLE_CUSTOMERS 14=LIQUIDITY_PROVIDER 15=OPERATING 16=CLEARING_FUND 17=FUTURES_MARKET_MAKER
582	CustOrderCapacity	N	Int	1=OWN_ACCOUNT 2=PROPRIETARY_ACCOUNT 3=FINANCIAL_ADVISOR 4=ALL_OTHER 5=RETAIL_CUSTOMER
453	NoPartyIDs	N	NumingGroup	Number of PartyID (448), PartyIDSource (447), and PartyRole (452) entries
→ 448	PartyID	N	String	Party Identifier/Code
→ 447	PartyIDSource	N	Char	D=Proprietary
→ 452	PartyRole	N	Int	1=EXECUTING_FIRM 3=CLIENT_ID 24=CUSTOMER_ACCOUNT
828	TrdType	N	int	Present on trades only. 0 = Regular trade
880	TrdMatchID	C	String	Always populated for trades. Note that buyer and seller will receive the same value. Will match the *TradeID (1003)* value on market data updates. Typically a 13-character alphanumeric string.
1028	ManualOrderIndicator	N	Boolean	Indicates if the order was initially received manually (as opposed to electronically)

Tag	Field Name	Req'd	Data Type	Comments
1057	AggressorIndicator	C	Boolean	Always populated for trades. Identifies whether this order was the aggressor in the trade.
378	ExecRestatementReason	N	int	Provided for unsolicited cancels. For an admin-initiated cancel, cancel on disconnect, or automatic order derisking: 8 = Market / exchange option For self-match prevention: 99 = Other
110	MinQty	N	Qty	Minimum required execution quantity for the order (if specified)
6127	ConditionTriggerMethod	N	int	The reference price used for triggering the stop order. 2 = Last price 5 = Settlement price
7928	SelfMatchPreventionID	N	String	Unique identifier for the self-match prevention instruction.
8000	SelfMatchPreventionInstruction	N	String	Self-match instruction. O = Cancel oldest (resting) order N = Cancel newest (aggressive) order
< Standard Trailer >		Y		

Figure 5: Simple limit order for 150 is acknowledged, partially filled for 100, and fully filled



Example 6: Acknowledgement of new limit order to buy 1,000 shares at 50.00

```
8=FIXT.1.1 | 9=269 | 35=8 | 34=14 | 49=TARGET | 52=20240517-19:00:28 | 56=SENDER |
57=SENDERSUB | 1=ACCT | 6=0.00 | 11=1182560819 | 14=0 | 17=1HPT7DPFMC5KW | 22=8 |
31=0.00 | 32=0 | 37=1HQ4A5T0EDM00 | 38=1000 | 39=0 | 40=2 | 44=50.00 | 48=GOOG |
54=1 | 55=GOOG | 59=0 | 60=20240517-19:00:28.678960817 | 99=0.00 | 150=0 |
151=1000 | 581=3 | 582=1 | 10=088 |
```

Example 7: Execution of 500 shares at a price of 50.00

```
8=FIXT.1.1 | 9=338 | 35=8 | 34=33 | 49=TARGET | 52=20240517-19:06:47.985808615 |
56=SENDER | 57=SENDERSUB | 1=ACCT | 6=50.00 | 11=1182560826 | 14=500 |
17=1HPT7DPFMC5M5 | 22=8 | 31=50.00 | 32=500 | 37=1HQ4A5T0EDM07 | 38=500 | 39=2 |
40=2 | 44=50.00 | 48=GOOG | 54=2 | 55=GOOG | 59=0 | 60=20240517-19:06:47.977567695
| 99=0.00 | 119=25000.00 | 150=F | 151=0 | 381=25000.00 | 581=3 | 582=1 | 828=0 |
880=1HPT7DPFMC5M4 | 1057=Y | 10=116 |
```

Example 8: Expiry of FOK order (without execution)

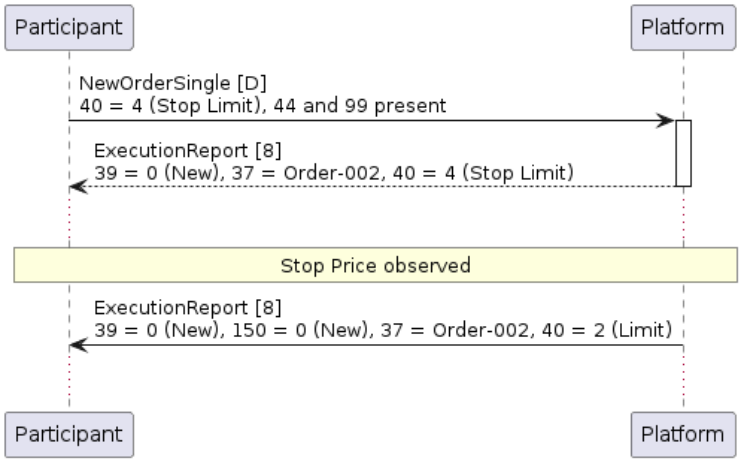
```
8=FIXT.1.1 | 9=275 | 35=8 | 34=43 | 49=TARGET | 52=20240517-19:09:23.494862156 |
56=SENDER | 57=SENDERSUB | 1=ACCT | 6=0.00 | 11=1182560830 | 14=0 |
17=1HPT7DPFMC5MB | 22=8 | 31=0.00 | 32=0 | 37=1HQ4A5T0EDM0A | 38=500 | 39=C | 40=2
| 44=50.01 | 48=GOOG | 54=2 | 55=GOOG | 59=4 | 60=20240517-19:09:23.491276593 |
99=0.00 | 150=C | 151=0 | 581=3 | 582=1 | 10=201 |
```

Triggering Stop Orders

Stop Orders which have been accepted by the Platform are immediately acknowledged with an ExecutionReport [8] message indicating *OrdStatus (39)* = 0 (New).

They remain outside the central limit order book until the *StopPx (99)* has been observed, triggering the release of either a Limit or Market-to-Limit order into the order book. At this point, the Platform will send a second, unsolicited Execution Report [8] with a matching *OrderID (37)* value, and the *OrdType (40)* of either 2 (Limit) or K (market-to-limit). The *OrdStatus (39)* of this triggered order will be 0 (New).

Figure 6: Triggering of Stop Limit order



Example 9: Entry of a Stop Limit order

```
8=FIXT.1.1 | 9=160 | 35=D | 49=SENDER | 56=TARGET | 34=119 | 52=20240521-09:45:21
| 11=1886428727 | 21=1 | 55=GOOG | 54=1 | 60=20240521-09:45:21 | 40=4 | 44=0.03 |
38=1500 | 50=SENDERSUB | 1=ACCT | 59=0 | 99=0.03 | 10=163 |
```

Example 10: Initial acknowledgement of Stop Limit order

```
8=FIXT.1.1 | 9=279 | 35=8 | 34=112 | 49=TARGET | 52=20240521-09:45:21.252049258 |
56=SENDER | 57=SENDERSUB | 1=ACCT | 6=0.00 | 11=1886428727 | 14=0 |
17=1HPT7DQ1GC4C4 | 22=8 | 31=0.00 | 32=0 | 37=1HQ4A5T0EDM1T | 38=1500 | 39=0 |
40=4 | 44=0.03 | 48=GOOG | 54=1 | 55=GOOG | 59=0 | 60=20240521-09:45:21.246689976
| 99=0.03 | 150=0 | 151=1500 | 581=3 | 582=1 | 10=079 |
```

Example 11: ExecutionReport indicating conversion of Stop LImit into Limit order

```
8=FIXT.1.1 | 9=293 | 35=8 | 34=126 | 49=TARGET | 52=20240521-09:52:30.011976583 |
56=SENDER | 57=SENDERSUB | 1=ACCT | 6=0.00 | 11=1886428732 | 14=0 |
17=1HPT7DQ1GC4ET | 22=8 | 31=0.00 | 32=0 | 37=1HQ4A5T0EDM1T | 38=1500 | 39=0 |
40=2 | 41=1886428727 | 44=0.03 | 48=GOOG | 54=1 | 55=GOOG | 59=0 | 60=20240521-
09:52:30.004561670 | 99=0.02 | 150=0 | 151=1500 | 581=3 | 582=1 | 10=253 |
```

Market-To-Limit Order Behaviour

Market-to-limit orders (*OrderType (39)* = K) are unpriced orders which are designed to execute against any existing orders in the order book upon arrival, with any remaining order balance automatically converted into a Limit order at the last trade or “worst fill” price.

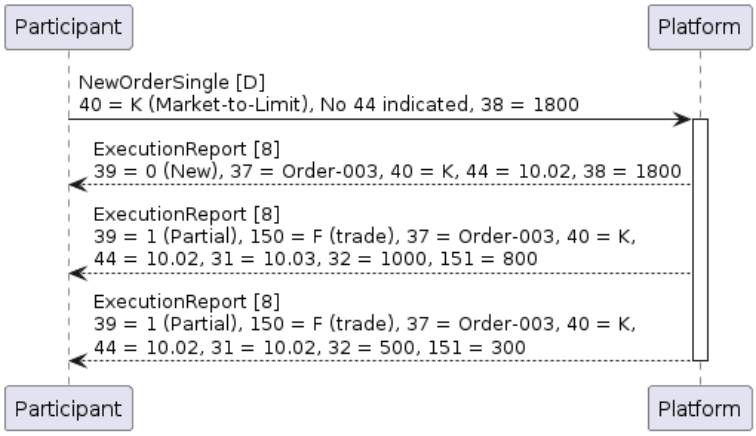
To illustrate this behavior, consider the below order book:

BUY		**SELL**	
Quantity	Price	Price	Quantity
1,000	10.03	10.04	2,500
500	10.02		

When a new Market-to-Limit order selling 1,800 hits the order book, then the order will immediately trade 1,000 shares at 10.03, and a further 500 shares at 10.02. Since there are no further bids, the Platform will place the remaining 300 shares into the order book at a price equal to the last fill price of 10.02.

BUY		**SELL**	
Quantity	Price	Price	Quantity
		10.02	300 (market-to-limit remainder)
		10.04	2,500

Figure 7: Handling of Market-to-Limit order (see example)



Note that unlike Stop Orders, Market-to-Limit orders retain the same *OrdType (40)* of K (Market-to-limit) throughout their life; the initial acknowledgement contains the limit *Price (44)* of the resting order.

Example 12: Initial acknowledgment of Market-to-Limit order indicating end Limit Price (44)

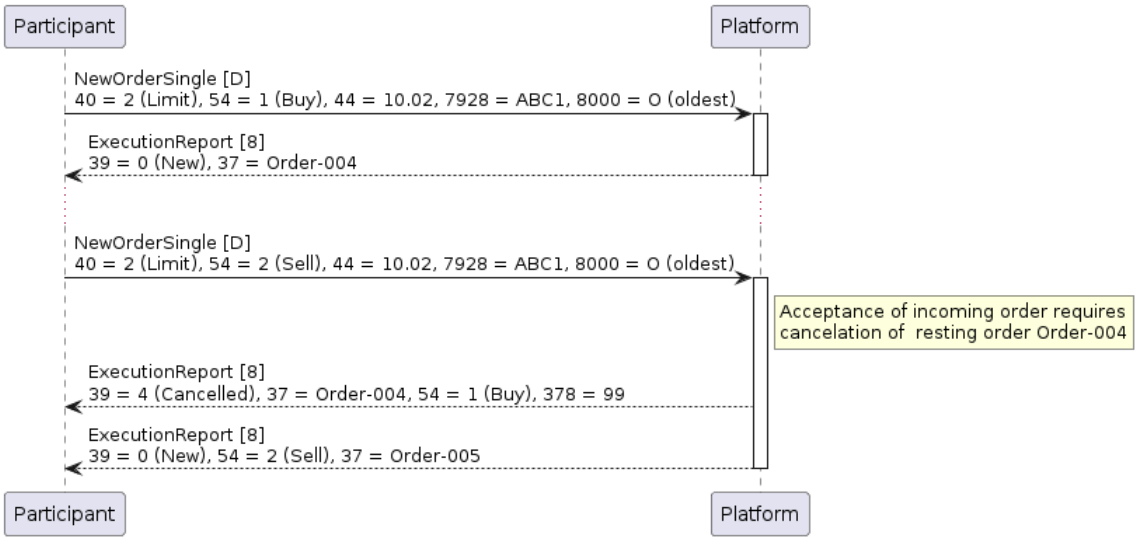
```
8=FIXT.1.1 | 9=278 | 35=8 | 34=11 | 49=TARGET | 52=20240521-10:40:28.546598320 |
56=SENDER | 57=SENDERSUB | 1=ACCT | 6=0.00 | 11=1886428739 | 14=0 |
17=1HPT7DQ1GC4GH | 22=8 | 31=0.00 | 32=0 | 37=1HQ4A5T0EDM20 | 38=1000 | 39=0 |
40=K | 44=0.02 | 48=GOOG | 54=2 | 55=GOOG | 59=0 | 60=20240521-10:40:28.542693638
| 99=0.00 | 150=0 | 151=1000 | 581=3 | 582=1 | 10=012 |
```

Self-Match Prevention

Orders may have self-match prevention enabled at either the FIX session level, or at an order-level basis using tags *SelfMatchPreventionID (7928)* and *SelfMatchPreventionInstruction (8000)*.

Where specified, two otherwise-executable orders from the same participant and which carry the same *SelfMatchPreventionID (7928)* will be prevented from matching by expiring one of the orders. Whether the resting or aggressive order is canceled is governed by *SelfMatchPreventionInstruction (8000)* **of the incoming order**.

Figure 8: Self-match prevention expires the oldest order to prevent self-trading



Note that resting order(s) will expire just after the incoming order has been accepted by the Platform, and before any trades have taken place. The incoming order is therefore permitted to trade against other orders in the order book as it entered the market.

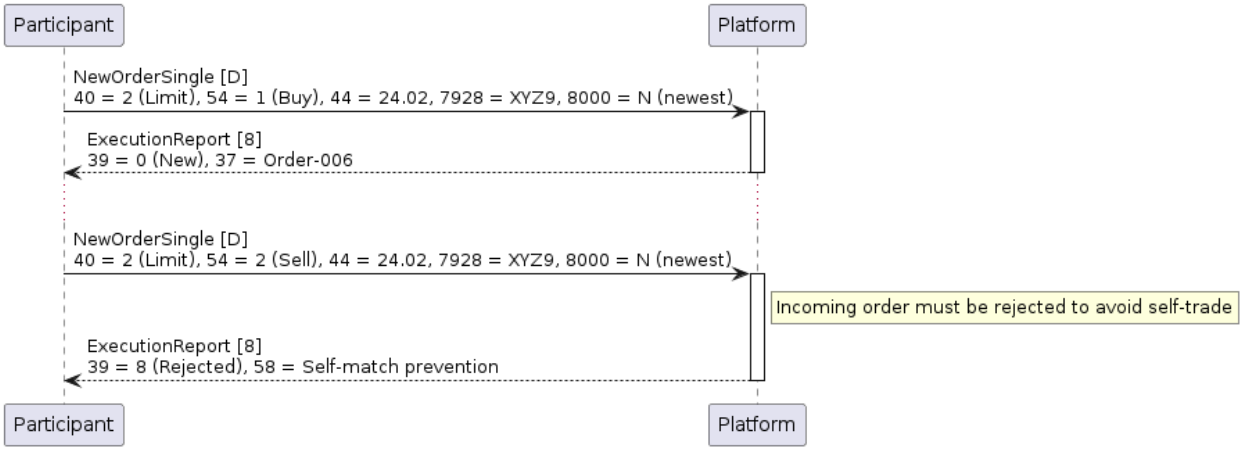
Example 13: ExecutionReport indicating resting order expiry as a result of self-match prevention

```
8=FIXT.1.1 | 9=341 | 35=8 | 34=69 | 49=TARGET | 52=20240521-11:22:26.078209896 |
56=SENDER | 57=SENDERSUB | 1=ACCT | 6=0.00 | 11=1886428747 | 14=0 |
17=1HPT7DQ1GC4JA | 22=8 | 31=0.00 | 32=0 | 37=1HQ4A5T0EDM26 | 38=1 | 39=4 | 40=2 |
41=1886428747 | 44=0.02 | 48=GOOG | 54=1 | 55=GOOG | 58=Self Match Prevention |
59=0 | 60=20240521-11:22:26.075068980 | 99=0.00 | 150=4 | 151=0 | 378=99 | 581=3 |
582=1 | 7928=111 | 8000=O | 10=039 |
```

If the *SelfMatchPreventionInstruction (8000)* is N (newest), then it is the incoming order which is rejected to prevent the execution, as shown below. This is the default behavior where *SelfMatchPreventionID (7928)* is present but *SelfMatchPreventionInstruction (8000)* is not specified.

Note that the incoming order is **fully canceled** in the case it would potentially match against other orders with the same *SelfMatchPreventionID (7928)*; partial execution against other orders is not permitted.

Figure 9: Self-match prevention causes newest order to be rejected



Example 14: ExecutionReport immediately rejecting incoming order as a result of self-match prevention

```
8=FIXT.1.1 | 9=314 | 35=8 | 34=109 | 49=TARGET | 52=20240521-11:55:30.495116582 |
56=SENDER | 57=SENDERSUB | 1=ACCT | 6=0.00 | 11=1886428755 | 14=0 |
17=1HPT7DQ1GC4JT | 22=8 | 31=0.00 | 32=0 | 37=1HQ4A5T0EDM2E | 38=10 | 39=8 | 40=2
| 44=0.02 | 48=GOOG | 54=1 | 55=GOOG | 58=Self Match Prevention | 59=0 |
60=20240521-11:55:30.487845738 | 99=0.00 | 103=0 | 150=8 | 151=0 | 581=3 | 582=1 |
7928=222 | 10=165 |
```

Order Modification

Participants may request to amend any open order using the OrderCancelReplaceRequest [G] message.

In this case, the order to be amended is identified using the last-entered *ClOrdID (11)* value relating to the order into the *OrigClOrdID (41)* field. Following FIX convention, this modification message will also contain a fresh *ClOrdID (11)* value which can then be used by the Participant to further modify the order if required. Note that the FIGURE Platform does **not** require the entry of the (platform-generated) *OrderID (37)* in order to amend the order.

Table 17: OrderCancelReplaceRequest [G] message

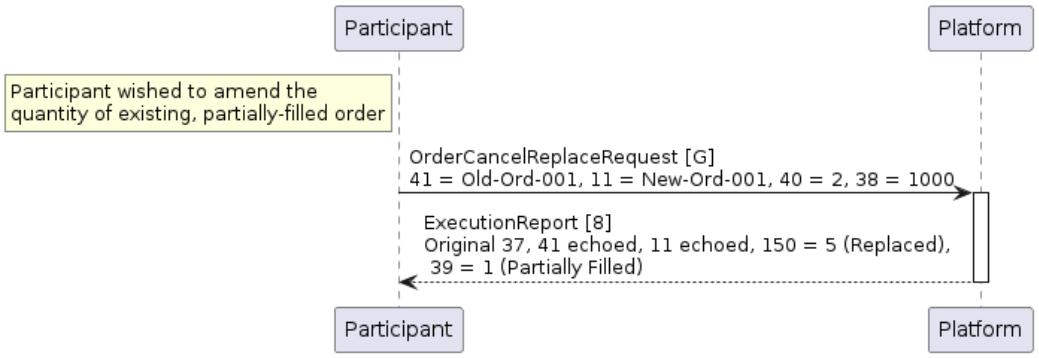
Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = G
11	ClOrdID	Y	String	Fresh, participant-generated, unique reference for this message. Must be different from the original *ClOrdID (11)* for the order.
22	SecurityIDSource	C	String	Required if *SecurityID (48)* is provided. The only acceptable value is 8 (Exchange Symbol), even if *security_id_source* on the FIGURE instrument is set to something other than Exchange Symbol.
41	OrigClOrdID	Y	String	The last (participant-generated) *ClOrdID (11)* reference for the order. This might be from the original NewOrderSingle (first amend), or prior amends.
48	SecurityID	C	String	Must match the original order (no amends). Instrument symbol. Required if *SecurityIDSource (22)* is provided. Either *SecurityID (48)* or *Symbol (55)* must be provided. Even if *security_id* on the FIGURE instrument is defined as something other than the FIGURE instrument symbol, this still must be set to the instrument symbol.
1	Account	N	String	Account reference as declared to the exchange operator.
18	ExecInst	N	MultipleChar	Must match the original order (no amends) G = All or None c = Ignore Price Validity Checks (only for market-to-limit orders to sell) 6 = Participate Don't Initiate
110	MinQty	N	Qty	Must match the original order (no amends)
55	Symbol	C	String	Must match the original order (no amends). Instrument symbol. Either *SecurityID (48)* or *Symbol (55)* must be provided.
38	OrderQty	C	Qty	Updated total order quantity. Only required if amending the quantity. Note that if the order has already been partially filled, then this still refers to the overall order quantity (i.e. not the remaining

Tag	Field Name	Req'd	Data Type	Comments
				quantity required). Can not be amended below *CumQty (14)*.
44	Price	N	Price	Updated order price. Required for orders where *OrdType (40)* = 2 (Limit).
99	StopPx	N	Price	Updated stop order price. Required for orders where *OrdType (40)* = 2 (Limit).
59	TimeInForce	N	char	0 = Good for day [Default] 1 = Good till cancel 3 = Immediate or cancel 4 = Fill or kill 6 = Good till date
126	ExpireTime	N	UTCTime	Required when *TimeInForce (59)* = Good Till Date (6). Order expiry time in UTC.
167	SecurityType	N	String	Type of instrument CS = Common Stock EVENT = Event contract FUT = Future FXSPOT = FX Spot (incl. crypto) FXSWAP = FX Swap MLEG = Multileg Instrument OOF = Options on Futures OPT = Option NONE = Other
762	SecuritySubType	N	String	Sub-type qualification/identification of the SecurityType <167> (e.g. for SecurityType="REPO"), or the CFICode <461> if SecurityType is not specified. If specified, SecurityType or CFICode is required.
1028	ManualOrderIndicator	N	Boolean	Indicates if the order was initially received manually (as opposed to electronically)
7928	SelfMatchPreventionID	N	String	Unique identifier to be returned in the case of a self-match prevention cancellation.
8000	SelfMatchPreventionInstruction	C	String	Self-match instruction. O = Cancel oldest (resting) order N = Cancel newest (aggressive) order
< Standard Trailer >		Y		

If the order has been successfully modified, the FIGURE Platform will respond with an ExecutionReport [8] message which echoes many of the key (updated) order attributes, and with *ExecType (150)* = 5 (replaced).

Note that a modification which **increases** the remaining order quantity will lose its time priority at a price level. Any modifications to **reduce** order quantity (or other non-price modifications which do not change quantity) will not cause the order to lose their order book priority.

Figure 10: Successful amend of existing order to adjust OrderQty [38]



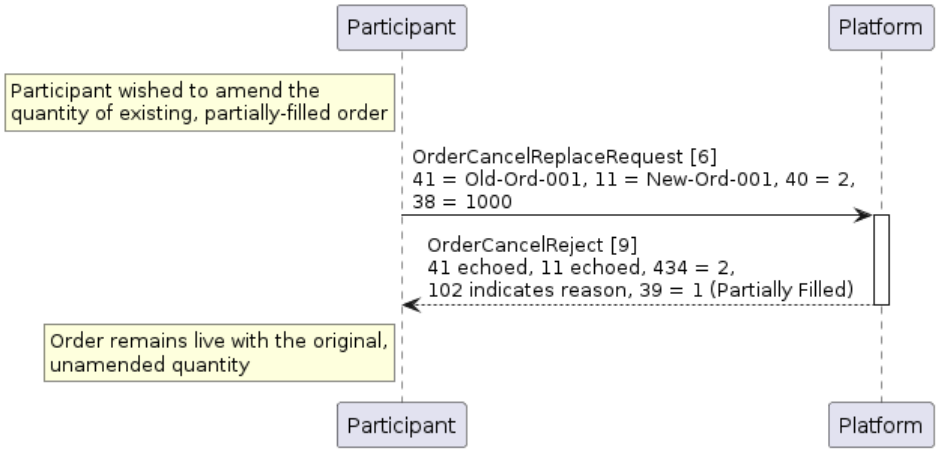
Example 15: Request to replace an existing order

```
8=FIXT.1.1 | 9=127 | 35=G | 49=SENDER | 56=TARGET| 34=66 | 52=20240517-19:17:08 |
50=SENDERSUB | 41=1182560827 | 11=1182560836 | 55=GOOG | 54=2 | 40=2 | 38=500 |
44=1000 | 10=061 |
```

If the order can not be modified for any reason (for example if the requested price / order size are not acceptable, or the order has already expired), then the FIGURE Platform will respond with an OrderCancelReject [9] message indicating the reason.

Where the amendment has been rejected, the existing order remains working with prior attributes.

Figure 11: Unsuccessful amend of existing order to adjust OrderQty [38]



Order Cancellation

Participants may request to cancel any open order using the OrderCancelRequest [F] message.

As with order amend requests, the order to be canceled is identified using the last-entered *ClOrdID (11)* value relating to the order into the *OrigClOrdID (41)* field.

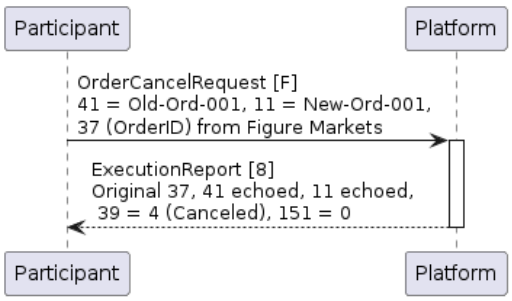
If the order has been canceled, the FIGURE Platform will respond with an ExecutionReport [8] message which echoes many of the key (updated) order attributes, and with *ExecType (150)* = 4 (canceled).

Table 18: OrderCancelRequest [F] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = F
11	ClOrdID	Y	String	Fresh, participant-generated, unique reference for this OrderCancelRequest. Must be different from the original *ClOrdID (11)* for the order.
22	SecurityIDSource	C	String	Required if *SecurityID (48)* is provided. The only acceptable value is 8 (Exchange Symbol), even if *security_id_source* on the FIGURE instrument is set to something other than Exchange Symbol.
41	OrigClOrdID	Y	String	The last (participant-generated) *ClOrdID (11)* reference for the order. This might be from the original NewOrderSingle or prior amends.
48	SecurityID	C	String	Instrument symbol as indicated on the order. Required if *SecurityIDSource (22)* is provided. Either *SecurityID (48)* or *Symbol (55)* must be provided. Even if *security_id* on the FIGURE instrument is defined as something other than the FIGURE instrument symbol, this still must be set to the instrument symbol.
55	Symbol	C	String	Instrument symbol as indicated on the order. Either *SecurityID (48)* or *Symbol (55)* must be provided.
< Standard Trailer >		Y		

If the request to cancel the order passed validation, then the FIGURE Platform will acknowledge the order cancellation with an ExecutionReport [8] message indicating *OrdStatus (39)* = 4 (canceled), and *LeavesQty (151)* = 0.

Figure 12: Successful cancelation of existing order



Example 16: Example of order cancel message

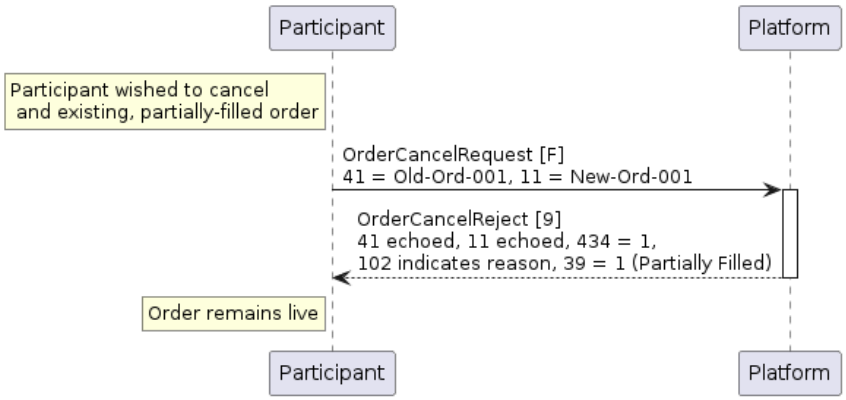
```
8=FIXT.1.1 | 9=107 | 35=F | 49=SENDER |  56=TARGET | 34=96 | 52=20240517-19:29:50
| 50=SENDERSUB | 41=1182560834 | 11=1182560840 | 55=GOOG | 54=1 | 10=244 |
```

Rejected attempts to cancel orders for any reason (for example order is no longer alive) are rejected using the OrderCancelReject [9] message, which identifies the reason for the rejection.

Table 19: OrderCancelReject [9] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 9
11	ClOrdID	Y	String	Echoed from the OrderCancelRequest
37	OrderID	Y	String	Will be 'NONE' for unknown orders
39	OrdStatus	Y	char	Status of the cancellation request. 8 = Rejected
41	OrigClOrdID	Y	String	Echoed from the OrderCancelRequest
58	Text	Y	String	Free text string providing additional rejection reason
102	CxlRejReason	Y	int	Reason for the rejection. 1 = Unknown order 2 = Exchange option 25 = Insufficient credit limit
434	CxlRejResponseTo	Y	char	1 = Order cancel request 2 = Order amend request
< Standard Trailer >		Y		

Figure 13: Unsuccessful attempt to cancel existing order



Example 17: Rejection of an unsuccessful cancel attempt

```
8=FIXT.1.1 | 9=145 | 35=9 | 34=91 | 49=TARGET | 52=20240521-09:26:30.378549737 |
56=SENDER | 57=SENDERSUB | 11=1886428723 | 37=NONE | 39=8 | 41=1886428676 |
58=Unknown order | 102=1 | 434=1 | 10=198 |
```

Participants may request to cancel multiple open orders using the OrderMassCancelRequest [q] message.

Multiple orders can be cancelled for a FIX session as outlined by the following table:

Table 20: Mass Cancellation Scenarios

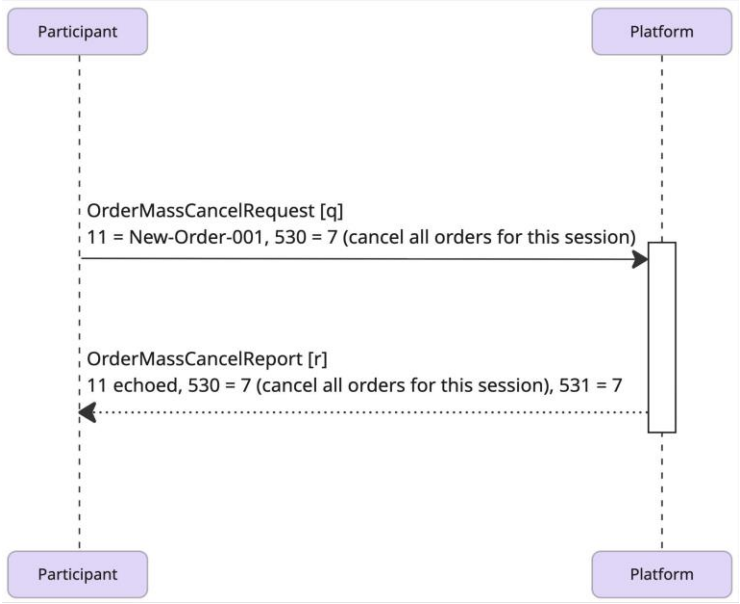
Mass Cancel Type	**MassCancelRequestType (530)**	**Required Filters**
By instrument	Cancel Orders for a Security (1)	*Symbol (55)* or *SecurityID (48)* + *SecurityIDSource (22)*
By account	Cancel All Orders (7)	*Account (1)*
By account + ClOrdLinkID	Cancel All Orders (7)	*Account (1)* and *ClOrdLinkID (583)*
Cancel all orders for the FIX session	Cancel All Orders (7)	Do not provide any filters

Table 21: OrderMassCancelRequest [q] message

Tag	**Field Name**	**Req'd**	**Data Type**	**Comments**
< Standard Header >		Y		35 = q

Tag	Field Name	Req'd	Data Type	Comments
1	Account	N	String	Account reference if mass cancel is by account or account + ClOrdLinkID
11	ClOrdID	Y	String	Unique, participant-created identifier for the OrderMassCancelRequest
22	SecurityIDSource	C	String	Required if *SecurityID (48)* is provided. The only acceptable value is 8 (Exchange Symbol), even if *security_id_source* on the FIGURE instrument is set to something other than Exchange Symbol.
48	SecurityID	C	String	Instrument symbol if mass cancel is by symbol. Either *SecurityID (48)* or *Symbol (55)* must be provided if *MassCancelRequestType (530)* is 7 (symbol). Required if *SecurityIDSource (22)* is provided. Even if *security_id* on the FIGURE instrument is defined as something other than the FIGURE instrument symbol, this still must be set to the instrument symbol.
55	Symbol	C	String	Instrument symbol if mass cancel is by symbol. Either *SecurityID (48)* or *Symbol (55)* must be provided if *MassCancelRequestType (530)* is 7 (symbol).
530	MassCancelRequestType	Y	char	1 = Cancel orders for a security 7 = Cancel all orders
583	ClOrdLinkID	N	String	Permits order originators to tie together groups of orders in which trades resulting from orders are associated for a specific purpose, for example the calculation of average execution price for a customer or to associate lists submitted to a broker as waves of a larger program trade.
< Standard Trailer >		Y		

Figure 14: Successful cancel of all open orders



Example 18: Example of mass order cancel message for open orders

```
8=FIXT.1.1 | 9=97 | 35=q | 49=SENDER | 56=TARGET | 34=101 | 52=20251216-19:29:50 |
11=New-Order-001 | 530=7 | 60=20251216-19:29:50 | 10=099 |
```

Table 22: OrderMassCancelReport [r] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = r
11	ClOrdID	Y	String	Echoed from the OrderMassCancelRequest
22	SecurityIDSource	Y	String	The *security_id_source* on the FIGURE instrument, if defined. If undefined, this is set to 8 (Exchange Symbol). 1 = CUSIP 2 = SEDOL 3 = QUIK 4 = ISIN 5 = RIC 6 = ISO Currency Code 7 = ISO Country Code 8 = Exchange Symbol 9 = Consolidated Tape Association A = Bloomberg Symbol B = Wertpapier C = Dutch D = Valoren

Tag	Field Name	Req'd	Data Type	Comments
				E = Sicovam F = Belgian G = Common H = Clearing House I = ISDA FpML Product Specification J = Option Price Reporting Authority K = ISDA FpML Product URL L = Letter of Credit M = Marketplace Assigned Identifier N = Markit RED Entity CLIP P = Markit RED Pair CLIP Q = CFTC Commodity Code R = ISDA Commodity Reference Price S = Financial Instrument Global Identifier (FIGI) T = Legal Entity Identifier (LEI) U = Synthetic V = Fidessa Instrument Mnemonic W = Index Name X = Uniform Symbol Y = Digital Token Identifier
48	SecurityID	Y	String	The *security_id* on the FIGURE instrument, if defined. If undefined this will match *Symbol (55).*
55	Symbol	Y	String	Instrument symbol
58	Text	N	String	Used to explain the reason for a mass cancel reject
60	TransactTime	Y	UTCTimestamp	Time the OrderMassCancelRequest was initiated
530	MassCancelRequestType	Y	char	1 = Cancel orders for a security 7 = Cancel all orders
531	MassCancelResponse	Y	char	0 = Cancel Request Rejected - See *MassCancelRejectReason (532)* 1 = Cancel orders for a security 7 = Cancel all orders
532	MassCancelRejectReason	Y	int	This is used in conjunction with a description of the reason for the rejection in tag 58 99 = Other
< Standard Trailer >		Y		

The OrderMassCancelReport [r] message is sent by the FIGURE Platform in response to an OrderMassCancelRequest [q]. Exactly one OrderMassCancelReport [r] is generated for each mass cancel request, regardless of the number of orders affected.

When a mass cancel request is accepted, the OrderMassCancelReport [r] will indicate success via the *MassCancelResponse (531)* field, whose value reflects the scope of the cancellation performed and mirrors the *MassCancelRequestType (530)* submitted on the request.

Individual order cancellations resulting from an accepted request are communicated separately using ExecutionReport [8] messages with *ExecType (150) = 4 (Canceled)*.

When a mass cancel request is rejected, the FIGURE Platform returns an OrderMassCancelReport [r] indicating rejection. In this case, *MassCancelResponse (531)* is set to 0 (Cancel Request Rejected), *MassCancelRejectReason (532)* is populated, and *Text (58)* provides a human-readable explanation of the rejection. When a mass cancel request is rejected, no ExecutionReport [8] messages are generated for order cancellations.

Certain mass cancel requests require additional filters in order to be valid.

- When canceling all orders using a *ClOrdLinkID (583)*, *Account (1)* must also be supplied because ClOrdLinkID values are not globally unique across accounts.
    - If not provided, this results in an OrderMassCancelReport [r] indicating rejection, as described above
- Requests to cancel orders for a security require either *Symbol (55)* or *SecurityID (48)* + *SecurityIDSource (22)* to be provided.
    - If neither is provided, this results in a BusinessMessageReject [j] message

Example 19: Rejection of a mass order cancel request where a ClOrdLinkID was provided without an Account

```
8=FIXT.1.1 | 9=162 | 35=r | 49=TARGET | 56=SENDER | 34=102 |
52=20251216-19:29:51 | 11=EXAMPLE_ORDER_001 | 530=7 | 531=0 | 532=0 |
58=at least one account must be provided when ClOrdLinkID is provided | 10=033 |
```

Trade Busts

When the exchange busts a trade, the FIGURE Platform will send an ExecutionReport [8] message for each trade side that echoes the order attributes but with *ExecType (150)* = H (trade cancel).

Note that the order attributes reflect the values at the time the trade occurred, not those from the latest state of the order.

Certain attributes of interest on the bust Execution Report are highlighted in Table 23 below.

Table 23: ExecutionReport [8] message sent for a busted trade

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 8
6	AvgPx	Y	Price	The volume-weighted average price of all trades against this order as recorded at the time of the trade.
11	ClOrdID	Y	String	The participant-assigned ClOrdID value as of the trade.
14	CumQty	Y	Qty	Cumulative quantity on the order as of the trade.
17	ExecID	Y	String	Unique identifier for the bust Execution Report as assigned by FIGURE. Typically a 13-character alphanumeric string.
19	ExecRefID	Y	String	The identifier for the Execution Report of the trade that was busted.
31	LastPx	Y	Price	Price of the trade.
32	LastQty	Y	Qty	Quantity of the trade.
37	OrderID	Y	String	Unique identifier for the order as assigned by FIGURE. Typically a 13-character alphanumeric string.
39	OrdStatus	Y	char	The status of the order as of the trade. 1 = Partially filled 2 = Fully filled
54	Side	Y	char	1 = Buy 2 = Sell
58	Text	Y	String	The reason for the trade bust as given by the exchange.
60	TransactTime	Y	UTCTime	Timestamp when the bust occurred.
119	SettlCurrAmt	Y	Amt	Total amount of the trade. Equal to *LastPx (31)* x *LastQty (32)*
150	ExecType	Y	char	H = Trade Cancel
151	LeavesQty	Y	Qty	Remaining, unexecuted quantity left on the order as of the trade.
381	GrossTradeAmt	Y	Amt	Total amount traded across all fills for this order as of the trade. Equal to *AvgPx (6)* x *CumQty (38)*.
880	TrdMatchID	Y	String	The trade identifier as assigned by FIGURE. Will match the *TradeID (1003)* value on

Tag	Field Name	Req'd	Data Type	Comments
				market data updates. Typically a 13-character alphanumeric string.
< Standard Trailer >		Y		

Market & Reference Data

The market data sessions for the FIGURE Platform are available by a second, separate FIX gateway; they are not accessible via the order-entry session.

When displaying the order book, the FIGURE Platform provides a Market-by-Order view; i.e each order is displayed individually with a corresponding timestamp (used to determine time priority within a price level). Each order also carries the unique OrderID reference, which Participants can use to identify their own orders in market data.

Subscribing to Market Data

Participants can subscribe to market data for a given symbol using a MarketDataRequest [V] message.

Table 24: MarketDataRequest [V] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = V
262	MDReqID	Y	String	Unique ID for this request
263	SubscriptionRequest Type	Y	char	Type of subscription requested 0 = Snapshot 1 = Snapshot plus Updates 2 = Delete previous request (unsubscribe)
264	MarketDepth	Y	int	Depth requested. (Maximum of 25 levels) 0 = Full book depth 1 = Top of book (best prices only) 2+ = Number of levels requested
267	NoMDEntryTypes	N	NumInGroup	
→ 269	MDEntryType	N	char	A repeating group of MD Entry Types requested 0 = Bid 1 = Offer 2 = Trade 4 = Opening Price 5 = Closing Price 6 = Settlement Price

Tag	Field Name	Req'd	Data Type	Comments
				7 = Trading Session High Price 8 = Trading Session Low Price B = Trade Volume C = Open Interest g = Trading Reference Price
146	NoRelatedSym	Y	NumInGroup	Number of symbols requested
→ 22	SecurityIDSource	C	String	Required if *SecurityID (48)* is provided. The only acceptable value is 8 (Exchange Symbol), even if *security_id_source* on the FIGURE instrument is set to something other than Exchange Symbol.
→ 48	SecurityID	C	String	Instrument symbol. Required if *SecurityIDSource (22)* is provided. Either *SecurityID (48)* or *Symbol (55)* must be provided. Even if *security_id* on the FIGURE instrument is defined as something other than the FIGURE instrument symbol, this still must be set to the instrument symbol.
→ 55	Symbol	C	String	Instrument symbol. Either *SecurityID (48)* or *Symbol (55)* must be provided.
< Standard Trailer >		Y		

If the MarketDataRequest [V] message is valid, FIGURE Platform will respond with a single MarketDataSnapshotFullRefresh [W] message for each requested Instrument, providing details of all orders in the order book (all levels as a repeating group within a single message). Note that the (repeating) *MDEntryType (269)* field can be specified if required to tailor the elements returned.

Example 20: Request a snapshot of all market data elements using MarketDataRequest [V] message

```
8=FIXT.1.1 | 9=92 | 35=V | 49=SENDER | 56=TARGET | 34=4 | 52=20240517-19:05:47 |
262=MD-REQ-001 | 263=1 | 264=3 | 146=1 | 55=GOOG | 10=075 |
```

Note that since the FIGURE Platform returns market data split by order, requesting only the best price using *MarketDepth (264)* = 1 (Top of book) may still return multiple bid and offer entries if there is more than one order at this price.

Table 25: MarketDataSnapshotFullRefresh [W] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = W

Tag	Field Name	Req'd	Data Type	Comments
22	SecurityIDSource	Y	int	The *security_id_source* on the FIGURE instrument, if defined. If undefined, this is set to 8 (Exchange Symbol). 1 = CUSIP 2 = SEDOL 3 = QUIK 4 = ISIN 5 = RIC 6 = ISO Currency Code 7 = ISO Country Code 8 = Exchange Symbol 9 = Consolidated Tape Association A = Bloomberg Symbol B = Wertpapier C = Dutch D = Valoren E = Sicovam F = Belgian G = Common H = Clearing House I = ISDA FpML Product Specification J = Option Price Reporting Authority K = ISDA FpML Product URL L = Letter of Credit M = Marketplace Assigned Identifier N = Markit RED Entity CLIP P = Markit RED Pair CLIP Q = CFTC Commodity Code R = ISDA Commodity Reference Price S = Financial Instrument Global Identifier (FIGI) T = Legal Entity Identifier (LEI) U = Synthetic V = Fidessa Instrument Mnemonic W = Index Name X = Uniform Symbol Y = Digital Token Identifier
48	SecurityID	Y	String	The *security_id* on the FIGURE instrument, if defined. If undefined this will match *Symbol (55).*
55	Symbol	Y	String	Instrument symbol
167	SecurityType	N	String	Type of instrument CS = Common Stock EVENT = Event contract FUT = Future FXSPOT = FX Spot (incl. crypto) FXSWAP = FX Swap IRS = Interest Rate Swap OOF = Options on Futures OPT = Option NONE = Other

Tag	Field Name	Req'd	Data Type	Comments
762	SecuritySubType	N	String	Sub-type qualification/identification of the SecurityType <167> (e.g. for SecurityType="REPO"), or the CFICode <461> if SecurityType is not specified. If specified, SecurityType or CFICode is required.
262	MDReqID	Y	String	The ID of the request as indicated on the request.
292	CorporateAction	C	char	Provided if the instrument is currently subject to a Corporate Action. A = Ex-Dividend B = Ex-Distribution C = Ex-Rights D = New E = Ex-Interest F = Cash Dividend G = Stock Dividend H = Non-Integer Stock Split I = Reverse Stock Split J = Standard-Integer Stock Split K = Position Consolidation L = Liquidation Reorganization M = Merger Reorganization N = Rights Offering O = Shareholder Meeting P = Spinoff Q = Tender Offer R = Warrant S = Special Action T = Symbol Conversion U = CUSIP or Name Change V = Leap Rollover W = Succession Event
268	NoMDEntries	Y	NumInGroup	The number of market data levels returned. Might be zero if the symbol is valid but there are currently no bids/offers in this symbol.
→ 269	MDEntryType	Y	char	Type of entry 0 = Bid 1 = Offer 2 = Trade 4 = Opening Price 5 = Closing Price 6 = Settlement Price 7 = Trading Session High Price 8 = Trading Session Low Price B = Trading Session Volume g = Trading Reference Price
→ 270	MDEntryPx	Y	Price	Price level
→ 271	MDEntrySize	C	Qty	Quantity of the individual order or trade, or the aggregate quantity where *MDEntryType (269)* = B (Trading Session Volume).

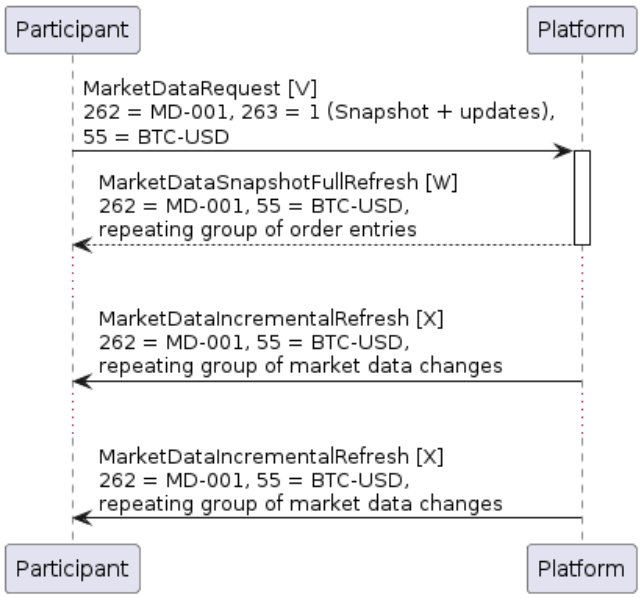
Tag	Field Name	Req'd	Data Type	Comments
				Not sent for session open / high / low.
→ 272	MDEntryDate	Y	UTCDateOnly	Time priority (date) of the order.
→ 273	MDEntryTime	Y	UTCTimeOnly	Time priority (time) of the order.
→ 336	TradingSessionID	C	String	Indicates the state of the instrument. Included for entries **other than** *MDEntryType (269) = 0 (Bid)* or *1 (Offer)*. ● CLOSED ● OPEN ● PREOPEN ● SUSPENDED ● EXPIRED ● TERMINATED ● HALTED ● MATCH_AND_CLOSE_AUCTION
→ 1151	SecurityGroup	C	String	The name of the group of related securities to which this instrument belongs.
→ 1070	MDQuoteType	C	int	Identifies market data quote type. Only sent for *MDEntryType (269)* = 4 (Opening Price). 0 = Indicative
→ 59	TimeInForce	C	char	Sent for *MDEntryType (269)* = 0 (Bid) or 1 (Offer). The time in force for this order. 0 = Good for day 1 = Good till cancel 6 = Good till date
→ 37	OrderID	C	String	Sent for *MDEntryType (269)* = 0 (Bid) or 1 (Offer). Matches the order ID in the ExecutionReport [8] acknowledgement, allowing Participants to identify their own orders within market data. Typically a 13-character alphanumeric string.
→ 278	MDEntryID	C	String	Sent for *MDEntryType (269)* = 0 (Bid) or 1 (Offer). Unique reference for the entry. Typically 13-character alphanumeric string.
→ 40	OrdType	C	char	Sent for *MDEntryType (269)* = 0 (Bid) or 1 (Offer). 2 = Limit order
→ 126	ExpireTime	N	UTCTimestamp	Sent for *MDEntryType (269)* = 0 (Bid) or 1 (Offer) where the order has *ExpiryTime (126)* set.
< Standard Trailer >		Y		

Example 21: Initial Market Data Snapshot (five repeating groups color-coded)

```
8=FIXT.1.1 | 9=458 | 35=W | 34=79 | 49=TARGET | 52=20240521-09:45:49.860198821 |
56=SENDER | 22=8 | 48=GOOG | 55=GOOG | 167=NONE | 262=1552371733 | 268=5 | 269=2 |
270=0.00 | 271=1499 | 272=20240521 | 273=09:06:39.324891684 | 336=OPEN | 269=4 |
270=3.00 | 272=20240515 | 273=21:24:03.898604733 | 336=OPEN | 1070=1 | 269=7 |
270=50.00 | 272=20240517 | 273=19:06:47.977567695 | 336=OPEN | 269=8 | 270=0.00 |
272=20240521 | 273=09:06:39.324891684 | 336=OPEN | 269=B | 270=93544.40 |
271=23645 | 272=20240521 | 273=09:06:39.324891684 | 336=OPEN | 1151=Equities |
10=199 |
```

Note the response will contain ONLY a snapshot of the current order book; it does not contain information about historic trades that have occurred on the platform.

Figure 15: Successful market data subscription with snapshot and incremental updates



If the participants subscribed to ongoing updates for the instrument(s), the platform will then start sending unsolicited MarketDataIncrementalRefresh [X] messages which contain a mixture of trade and/or order book update messages.

These incremental market data messages may contain a repeating group of updates in a single message. For example, an incoming sell order which enters the order book, executes against a resting buy order with the remainder written to the order book will trigger a MarketDataIncrementalRefresh [X] message containing the following updates:

1. The deletion of the previous best bid (as a result of the immediate fill),
2. The addition of a new best bid (next-best bid price),
3. The deletion of the best offer (new order has a better price),
4. The addition of a new best offer (representing the balance of the incoming sell order),

5. A record relating to the trade, and
6. A record relating to updated overall market volume

Table 26: MarketDataIncrementalRefresh [X] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = X
262	MDReqID	Y	String	The ID of the request as indicated on the request
268	NoMDEntries	Y	NumInGroup	
→ 279	MDUpdateAction	Y	char	The type of action conveyed by this block. 0 = New 1 = Change 2 = Delete
→ 269	MDEntryType	C	char	Type of entry. Set where *MDUpdateAction (279)* = (0) New or 1 (Change) 0 = Bid 1 = Offer 2 = Trade 4 = Opening Price 5 = Closing Price 6 = Settlement Price 7 = Trading Session High Price 8 = Trading Session Low Price B = Trade Volume g = Trading Reference Price
→ 278	MDEntryID	N	String	Unique reference for this entry. Typically a 13-character alphanumeric string.
→ 55	Symbol	Y	String	Instrument symbol
→ 22	SecurityIDSource	Y	int	The *security_id_source* on the FIGURE instrument, if defined. If undefined, this is set to 8 (Exchange Symbol). 1 = CUSIP 2 = SEDOL 3 = QUIK 4 = ISIN 5 = RIC 6 = ISO Currency Code 7 = ISO Country Code 8 = Exchange Symbol 9 = Consolidated Tape Association A = Bloomberg Symbol B = Wertpapier C = Dutch D = Valoren E = Sicovam F = Belgian

Tag	Field Name	Req'd	Data Type	Comments
				G = Common H = Clearing House I = ISDA FpML Product Specification J = Option Price Reporting Authority K = ISDA FpML Product URL L = Letter of Credit M = Marketplace Assigned Identifier N = Markit RED Entity CLIP P = Markit RED Pair CLIP Q = CFTC Commodity Code R = ISDA Commodity Reference Price S = Financial Instrument Global Identifier (FIGI) T = Legal Entity Identifier (LEI) U = Synthetic V = Fidessa Instrument Mnemonic W = Index Name X = Uniform Symbol Y = Digital Token Identifier
→ 48	SecurityID	Y	String	The *security_id* on the FIGURE instrument, if defined. If undefined this will match *Symbol (55)*.
→ 167	SecurityType	N	String	Type of instrument BASIS = Basis Swap CS = Common Stock EVENT = Event contract FRA = Forward Rate Agreement FSIRS = Forward Starting Interest Rate Swap FUT = Future FXSPOT = FX Spot (incl. crypto) FXSWAP = FX Swap OOF = Options on Futures OPT = Option NONE = Other
→762	SecuritySubType	N	String	Sub-type qualification/identification of the SecurityType <167> (e.g. for SecurityType="REPO"), or the CFICode <461> if SecurityType is not specified. If specified, SecurityType or CFICode is required.
→ 1151	SecurityGroup	N	String	Security sub-type. For example "Equities"
→ 270	MDEntryPx	N	Price	- Order level price where *MDEntryType (269)* = 0 (Bid) or 1 (Offer) - Traded price where *MDEntryType (269)* = 2 (Trade) - Total value traded where *MDEntryType (269)* = B (Trade Volume)
→ 271	MDEntrySize	N	Qty	- Remaining order size where *MDEntryType (269)* = 0 (Bid) or 1 (Offer).

Tag	Field Name	Req'd	Data Type	Comments
				Will be the quantity removed where *MDUpdateAction (279)* = 2 (Delete) - Trade size where *MDEntryType (269)* = 2 (Trade) - Total quantity traded where *MDEntryType (269)* = B (Trade Volume)
→ 272	MDEntryDate	N	UTCDateOnly	The date on which the price level or trade occurred
→ 273	MDEntryTime	N	UTCTimeOnly	The time at which the price level updated or trade occurred (in UTC)
→ 59	TimeInForce	N	char	Time in force for this order. 0 = Good for day 1 = Good till cancel 6 = Good till date
→ 126	ExpireTime	N	UTCTimestamp	Populated where *TimeInForce (59)* = 6 (Good Till Date)
→ 37	OrderID	N	String	Only sent for price level updates. Will match *OrderID (37)* in the ExecutionReport [8], allowing Participants to identify their own orders within market data.
→ 40	OrdType	N	char	2 = Limit order K = Market-to-limit order
→ 828	TrdType	N	int	Only sent for trades. 0 = Regular trade 1 = Block Trade 16 = All or None 99 = RFQ Trades
→ 1003	TradeID	N	String	Only sent for trade updates. Will match the *TrdMatchID (880)* in the ExecutionReport fill, allowing Participants to identify their own trades within market data.
→ 2446	AggressorSide	N	char	Only sent for trades. Indicates which side was the aggressor in a trade. 1 = Buy 2 = Sell
→ 336	TradingSessionID	C	String	Indicates the state of the instrument. Included for entries **other than** *MDEntryType (269) = 0 (Bid)* or *1 (Offer)*. ● CLOSED ● OPEN ● PREOPEN ● SUSPENDED ● EXPIRED ● TERMINATED ● HALTED

Tag	Field Name	Req'd	Data Type	Comments
				• MATCH_AND_CLOSE_AUCTION
< Standard Trailer >		Y		

Example 22 : Market Data incremental containing multiple updates (repeating groups color-coded)

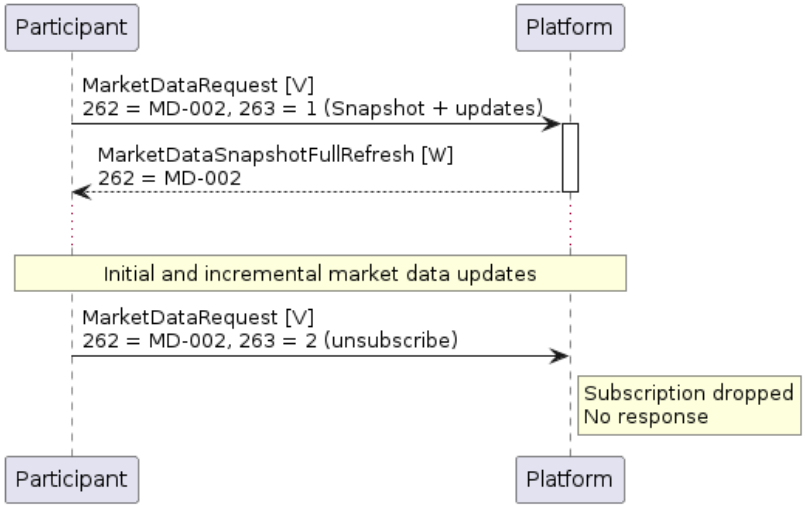
```
8=FIXT.1.1 | 9=987 | 35=X | 34=87 | 49=TARGET | 52=20240521-09:52:30.013930670 |
56=SENDER | 262=1552371733 | 268=6 | 279=0 | 269=0 | 278=1HQ4A5T0EDM1T | 55=GOOG |
48=GOOG | 22=8 | 167=NONE | 1151=Equities | 270=0.03 | 271=1500 | 272=20240521 |
273=09:52:30.004561670 | 59=0 | 37=1HQ4A5T0EDM1T | 40=2 | 279=0 | 269=1 |
278=1HQ4A5T0EDM1W | 55=GOOG | 48=GOOG | 22=8 | 167=NONE | 1151=Equities | 270=0.03
| 271=15 | 272=20240521 | 273=09:52:30.004561670 | 59=0 | 37=1HQ4A5T0EDM1W | 40=2
| 279=2 | 269=1 | 278=1HQ4A5T0EDM1W | 55=GOOG | 48=GOOG | 22=8 | 167=NONE |
1151=Equities | 270=0.03 | 271=0 | 272=20240521 | 273=09:52:30.004561670 | 59=0 |
37=1HQ4A5T0EDM1W | 40=2 | 279=2 | 269=0 | 278=1HQ4A5T0EDM1V | 55=GOOG | 48=GOOG |
22=8 | 167=NONE | 1151=Equities | 270=0.03 | 271=0 | 272=20240521 |
273=09:52:30.004561670 | 59=0 | 37=1HQ4A5T0EDM1V | 40=2 | 279=0 | 269=2 |
278=1HPT7DQ1GC4DS | 55=GOOG | 48=GOOG | 22=8 | 167=NONE | 1151=Equities | 270=0.03
| 271=15 | 272=20240521 | 273=09:52:30.004561670 | 59=0 | 40=2 | 828=0 |
1003=1HPT7DQ1GC4DS | 2446=2 | 279=0 | 269=B | 55=GOOG | 48=GOOG | 22=8 | 167=NONE
| 1151=Equities | 270=93544.85 | 271=23660 | 272=20240521 | 273=09:52:30.004561670
| 336=OPEN | 10=156 |
```

Unsubscribing from Market Data

Participants can unsubscribe from a piece of market data by entering a second MarketDataRequest [V] message, referencing the original *MDReqID (262)* reference and setting *SubscriptionRequestType (263)* = 2 (unsubscribe).

Note that there is no explicit response to requests to unsubscribe from the FIGURE Platform; a successful unsubscription will simply prevent further MarketDataIncrementalRefresh [X] messages.

Figure 16: Successful market data subscription and unsubscription



Reference Data

Participants may download a list of the instruments available (or reference data about them) to trade on the FIGURE Platform by submitting a SecurityListRequest [x] message.

Table 27: SecurityListRequest [x] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = x
320	SecurityReqID	Y	String	Unique ID associated with this request.
336	TradingSessionID	N	String	Used to filter for instruments of the specified market state: • CLOSED • OPEN • PREOPEN • SUSPENDED • EXPIRED • TERMINATED • HALTED • MATCH_AND_CLOSE_AUCTION
559	SecurityListRequestTyp e	Y	int	The type of request being made. 0 = Individual symbol 4 = All Securities
22	SecurityIDSource	C	String	Required if *SecurityID (48)* is provided. The only acceptable value is 8 (Exchange Symbol), even if *security_id_source* on the FIGURE instrument is set to something other than Exchange Symbol.

Tag	Field Name	Req'd	Data Type	Comments
48	SecurityID	C	String	Instrument symbol. Required if *SecurityIDSource (22)* is provided. Either *SecurityID (48)* or *Symbol (55)* must be provided if *SecurityListRequestType(559)* = 0 (individual security). Even if *security_id* on the FIGURE instrument is defined as something other than the FIGURE instrument symbol, this still must be set to the instrument symbol.
55	Symbol	C	String	Either *SecurityID (48)* or *Symbol (55)* must be provided if *SecurityListRequestType(559)* = 0 (individual security).
< Standard Trailer >		Y		

Example 23: Request for information on all securities

```
8=FIXT.1.1 | 9=75 | 35=x | 49=SENDER | 56=TARGET | 34=12 | 52=20240516-14:28:38 |
320=REF-DATA-001 | 559=4 | 10=048 |
```

Upon receipt, the FIGURE Platform will respond with a SecurityList [y] message containing the requested information. Note that the content provided by the SecurityList [y] message does not differ depending on whether an individual symbol was listed versus all securities.

Table 28: SecurityList [y] message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = y
320	SecurityReqID	Y	String	The unique *SecurityReqID (320)* sent on the request.
322	SecurityResponseID	Y	String	Unique ID for this response. Typically a 13 character alphanumeric string.
560	SecurityRequestResult	Y	int	The status of the request. 0 = Valid request 1 = Invalid or unsupported request 3 = Not authorized to retrieve instrument data
146	NoRelatedSym	C	NumInGroup	Number of instruments to be returned. Only present for valid requests.
→55	Symbol	Y	String	Instrument symbol

Tag	Field Name	Req'd	Data Type	Comments
→48	SecurityID	Y	String	The *security_id* on the FIGURE instrument, if defined. If undefined this will match *Symbol (55).*
→22	SecurityIDSource	Y	int	The *security_id_source* on the FIGURE instrument, if defined. If undefined, this is set to 8 (Exchange Symbol). 1 = CUSIP 2 = SEDOL 3 = QUIK 4 = ISIN 5 = RIC 6 = ISO Currency Code 7 = ISO Country Code 8 = Exchange Symbol 9 = Consolidated Tape Association A = Bloomberg Symbol B = Wertpapier C = Dutch D = Valoren E = Sicovam F = Belgian G = Common H = Clearing House I = ISDA FpML Product Specification J = Option Price Reporting Authority K = ISDA FpML Product URL L = Letter of Credit M = Marketplace Assigned Identifier N = Markit RED Entity CLIP P = Markit RED Pair CLIP Q = CFTC Commodity Code R = ISDA Commodity Reference Price S = Financial Instrument Global Identifier (FIGI) T = Legal Entity Identifier (LEI) U = Synthetic V = Fidessa Instrument Mnemonic W = Index Name X = Uniform Symbol Y = Digital Token Identifier
→460	Product	N	Int	Indicates the type of product the security is associated with. 1=AGENCY 2=COMMODITY 3=CORPORATE 4=CURRENCY 5=EQUITY 6=GOVERNMENT 7=INDEX 8=LOAN 9=MONEYMARKET 10=MORTGAGE 11=MUNICIPAL 12=OTHER

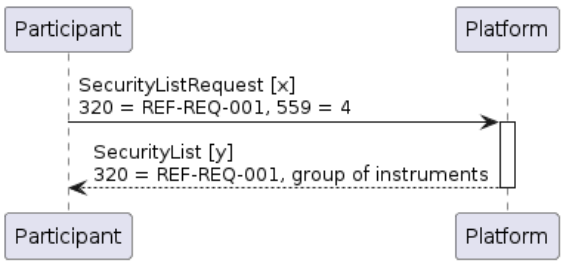
Tag	Field Name	Req'd	Data Type	Comments
				13=FINANCING 14=ENERGY
→167	SecurityType	N	String	Type of instrument CS = Common Stock EVENT = Event contract FUT = Future FXSPOT = FX Spot (incl. crypto) FXSWAP = FX Swap MLEG = Multileg Instrument OOF = Options on Futures OPT = Option NONE = Other
762	SecuritySubType	N	String	Sub-type qualification/identification of the SecurityType <167> (e.g. for SecurityType="REPO"), or the CFICode <461> if SecurityType is not specified. If specified, SecurityType or CFICode is required.
→231	ContractMultiplier	N	float	The ratio or multiplier to convert from "nominal" units (e.g. contracts) to total units (e.g. shares). Applicable for Fixed Income, Derivatives, etc.
→864	NoEvents	Y	NumInGroup	Number of repeating EventType entries. Will always be 1.
→→865	EventType	Y	int	Code to represent the type of event 5 = Activation
→→866	EventDate	Y	LocalMktDate	Date that the instrument first started (or will start) trading on the exchange in YYYYMMDD format.
→→ 868	EventText	Y	String	Event string. Always 'StartDate'.
→969	MinPriceIncrement	Y	float	Minimum price increment (tick size)
→1151	SecurityGroup	Y	String	The name of the group of securities to which this instrument belongs.
→562	MinTradeVol	Y	Qty	The minimum quantity allowed on an order. This field can be a decimal, indicating the ability to trade fractional shares of this instrument.
1306	PriceLimitType	N	Int	0 = Price 1 = Ticks 2 = Percentage
→ 1148	LowLimitPrice	N		

Tag	Field Name	Req'd	Data Type	Comments
→ 1149	HighLimitPrice	N		
→15	Currency	Y	Currency	Currency for the instrument, or quote currency in the case of Forex. For example, for BTC-USD, the Currency would be USD.
555	NoLegs	C		Number of <InstrumentLeg> repeating group instances. Required if 167=MLEG
→ 602	LegSecurityID	C	String	
→ 603	LegSecurityIDSource	C	String	
→ 609	LegSecurityType	C	String	
→ 611	LegMaturityDate	C	LocalMktDate	
→ 624	LegSide	C	Char	
→ 623	LegRatioQty	C	Float	
→ 764	LegSecuritySubType	N	String	Only used with 167/609=MLEG
< Standard Trailer >		Y		

Example 24: SecurityList [y] indicating two securities (color-coded)

```
8=FIXT.1.1 | 9=336 | 35=y | 34=9 | 49=TARGET | 52=20240516-14:28:38.864954771 |
56=SENDER | 146=2 | 55=GC-Dec-2030 | 48=GC-Dec-2030 | 22=8 | 167=NONE | 231=1 |
864=1 | 865=5 | 866=19700101 | 868=StartDate | 969=0.01 | 1151=GC | 562=1 | 15=USD
| 55=GOOG | 48=GOOG | 22=8 | 167=NONE | 231=1 | 864=1 | 865=5 | 866=19700101 |
868=StartDate | 969=0.01 | 1151=Equities | 562=1 | 15=USD | 320=2007026312 |
322=1HPT7F2AA6404 | 560=0 | 10=007 |
```

Figure 17: Request for security reference data



Appendix A: FIX Data Types

Type Name	Data Type	Description
int	Signed integer. Zeros permitted.	Sequence of digits without commas or decimals and optional sign character (ASCII characters - and 0 - 9). The sign character utilizes one byte (i.e. positive int is 99999 while negative int is -99999). Note that int values may contain leading zeros (e.g. 00023 = 23).
Length	Strictly positive integer.	int field representing the length in bytes.
SeqNum	Strictly positive integer.	int field representing a message sequence number.
NumInGroup	Strictly positive integer.	int field representing the number of entries in a repeating group.
float	Signed float, with optional decimal point.	The absence of the decimal point within the string will be interpreted as the float representation of an integer value. All float fields must accommodate up to fifteen significant digits. Float values may contain leading zeros (e.g. 00023.23 = 23.23) and may contain or omit trailing zeros after the decimal point (e.g. 23.0 = 23.0000 = 23 = 23.).
Qty	Positive float.	float field capable of storing either a whole number (no decimal places) of shares (securities denominated in whole units) or a decimal value containing decimal places for non-share quantity asset classes (securities denominated in fractional units).
Price	Signed float price.	float field representing a price with a varying number of decimal places. For certain asset classes prices may be negative values. For example, prices for options strategies can be negative under certain market conditions.
Amt	Signed float	float field typically representing a Price times a Qty.
char	Case-sensitive, single alphanumeric character	Can include any alphanumeric character or punctuation except the delimiter (SOH). All char fields are case sensitive (i.e. m != M).
Boolean	'Y' = True/Yes 'N' = False/No	char field containing one of two possible values.
String	Case-sensitive, alphanumeric string.	Can include any character or punctuation except the delimiter. All String fields are case sensitive (i.e. morstatt != Morstatt).

Type Name	Data Type	Description
MultipleCharValue	Space-delimited sequence of single character values	string field containing one or more space-delimited values (e.g. | 18=G c |).
Currency	Three character string.	string field representing a currency type using ISO 4217 Currency code (3 character) values.
UTCTimestamp	Formatted UTC timestamp (date and time)	string field representing time/date combination represented in UTC in either: - YYYYMMDD-HH:MM:SS (whole seconds) or - YYYYMMDD-HH:MM:SS.sss* format Where: - SS can be 60 seconds in the rare case of UTC leap second, and - sss* represents fractions of seconds, either - 3 digits to represent milliseconds, - 6 digits to convey microseconds, - 9 digits to convey nanoseconds, or - 12 digits to convey picoseconds.
UTCTimeOnly	Formatted UTC timestamp (time only)	string field representing a time represented in UTC in either: - HH:MM:SS (whole seconds) or - HH:MM:SS.sss* format
UTCDateOnly	Formatted UTC timestamp (date only)	string field representing Date represented in UTC in YYYYMMDD format.
LocalMktDate	Date in the timezone local to the sender	string field representing Date represented in sender's timezone in YYYYMMDD format.

Appendix B: List of FIX Messages

Appendix C: List of Supported Workflow Diagrams