ALPHAPOINT

OMS FIX Gateway

Version 1.1.0 | FIX 4.4 | Revised 2026-07-10

reconciled against the OMS QuickFIX Gateway implementation

Contents

Introduction

The OMS FIX Gateway provides programmatic access to the AlphaPoint OMS using the FIX Protocol for the purposes of entering orders and receiving executions. This document lays out the supported features, messages, and fields, and describes how to set up a FIX client to access them.

The gateway is implemented on the QuickFIX/N engine (`ThreadedSocketAcceptor`) and acts as a FIX **acceptor**; your client connects as an **initiator**.

FIX Versions Supported

The AlphaPoint OMS FIX Gateway supports **FIX Version 4.4** exclusively. The standard FIX 4.4 data dictionary is used (`APEXOMSQuickFIX/spec/FIX44.xml`).

Important Notes

- The system uses `SecurityID` **(48)** rather than `Symbol` (55) to identify an instrument. Always send `SecurityID`; `Symbol` is informational and ignored for order routing. The `<Symbol, SecurityID>` mapping is obtained from the `SecurityListRequest` / `SecurityList` workflow.
- The system only accepts a **64-bit integer** for `ClOrdID` (11) and `OrigClOrdID` (41). Non-numeric values are rejected with a `BusinessMessageReject`.
- **Authentication is Username/Password only.** There is **no API-key / HMAC-signature logon path** in this gateway. Supply your APEX username in `Username` (553) and password in `Password` (554) on the `Logon` message.
- **2FA must be disabled** for the trading user. FIX cannot deliver a 2FA challenge, and the gateway rejects logons for 2FA-enabled users.
- `RequestForPositions` **returns a snapshot only** — one `PositionReport` per product on the account. It does not stream position updates.
- Your **APEX Account ID is carried in** `SenderCompID` **(49)**, not in `Username`. See Session Identity & Comp IDs.

Prerequisites

Before connecting you need:

1. An **APEX user account** with email verified, not locked, and **2FA disabled**.
2. Your **APEX Account ID** (an integer) — this goes in `SenderCompID` (49). The account must be enabled for FIX trading (`account_config` key `OMS_FIX_GW_ENABLED = true`), and the user must be a member of that account.
3. The **gateway endpoint** — `host:port` of the FIX acceptor (default port `9821`).
4. The gateway's **server ID** (`ServerName`, e.g. `TESTOMSFIX` in dev or `APEXFIX-PROD` in production) — this goes in `TargetCompID` (56).
5. *(Optional)* a **TLS root** if the operator runs the acceptor with `UseSSL=true`.

> Note: an IP-whitelist feature exists (`account_config` key `FIX_IP_WHITELIST`), but QuickFIX/N 1.8.1 does not currently expose the remote IP to the gateway, so the whitelist check is inert in the current build (`DEV-9851`). Enforce source-IP restrictions at the network layer if required.

Session Identity & Comp IDs

Every FIX session represents **one APEX account**. The gateway derives your account from the CompIDs on your inbound messages:

Your outbound field	Value	Notes
`SenderCompID` (49)	**Your APEX Account ID** (integer, e.g. `48201`)	The gateway parses this as an `Int32` account id. If it is not an integer, application messages are rejected with `Invalid account`.
`TargetCompID` (56)	**The gateway's server ID** (`ServerName`, e.g. `TESTOMSFIX` / `APEXFIX-PROD`)	Provided by AlphaPoint.

Your outbound field	Value	Notes
`Username` (553)	Your APEX username	Authentication credential (separate from the account id).
`Password` (554)	Your APEX password	

On messages the gateway sends **to** you, these are reversed: `SenderCompID = <ServerName>` and `TargetCompID = <your AccountId>`. QuickFIX session strings therefore read `FIX.4.4:<ServerName>-><AccountId>` (e.g. `FIX.4.4:APEXFIX-PROD->48201`).

If you trade multiple accounts, open **one FIX session per account** (each account id is a distinct session).

Message Header / Trailer

Required-field indicator: **Y** = Yes, **N** = No, **Y/N** = Conditional (see Descr).

Header

Field	Required	Descr	Values / Formatting
BeginString [8]	Y	FIX version of the session.	`FIX.4.4`
BodyLength [9]	Y	Length of the message from the field after BodyLength up to (not including) the CheckSum field.	Integer
MsgType [35]	Y	The type of message.	See supported message types
SenderCompID [49]	Y	**Your APEX Account ID** (integer).	Integer
TargetCompID [56]	Y	The gateway's server ID (`ServerName`). Provided by AlphaPoint.	String
MsgSeqNum [34]	Y	Sequential message number; keep separate sequences for sent and received.	Integer
PossDupFlag [43]	N	Set on resent messages.	N = Original, Y = Possible Duplicate
PossResendFlag [97]	N	Message may contain info sent under another sequence number.	Y = Possible Resend, N = Original
SendingTime [52]	Y	Sending time of the message.	UTC Timestamp
OrigSendingTime [122]	Y/N	Required for messages resent in response to a ResendRequest — the original `SendingTime` .	UTC Timestamp

Trailer

Field	Required	Descr	Values / Formatting
CheckSum [10]	Y	Always the last field, followed by a single SOH (0x01). Computed over the entire message including the final delimiter before the checksum.	3-byte field

Application Messages (Client → Gateway)

Required-field indicator: **Y** = Yes, **N** = No, **Y/N** = Conditional (see Descr).

All application messages are gated on connection health and rate limits before processing. If the OMS path is unavailable, or a per-session rate limit is exceeded, the gateway responds with a `BusinessMessageReject` (see BusinessMessageReject [j]).

Logon [A]

The first message upon connecting. Failed logins are disconnected by the server (the gateway throws `RejectLogon`, which QuickFIX/N delivers to you as a `Logout`).

Authentication is **Username / Password only** — there is no API-key or signature-based logon.

Field	Required	Descr	Values / Formatting
HeartBtInt [108]	Y	Desired heartbeat interval in seconds. Suggested value 30.	Integer
ResetSeqNumFlag [141]	Y	Whether to reset sequence numbers for the session. Suggested `Y` on each new connection (the acceptor defaults to `ResetOnLogon=Y`).	Y/N
Username [553]	Y	Your APEX username.	String
Password [554]	Y	Your APEX password.	String

Logon reject reasons (returned in the `Logout` / `RejectLogon` text):

Reason text	Cause
`Max sessions reached`	The gateway's `MaxSessions` cap (default 100) is at capacity.
`Logon rate limit exceeded`	Per-IP logon throttle exceeded (5 burst, 1/sec). *(Currently inert — see the remote-IP note above.)*
`Logon failed`	Invalid credentials, user locked, email not verified, **2FA enabled**, or the account (`SenderCompID`) is not a FIX-enabled account for this user.

RequestForPositions [AN]

- Returns the current balance snapshot — one `PositionReport` (AP) per product on the account.
- Does **not** stream. Poll if you need updates, and be mindful of the overall rate limit.

Field	Required	Descr	Values / Formatting
PosReqID [710]	Y	Client-supplied request id; echoed on the `PositionReport`.	String
Account [1]	N	Ignored — the gateway uses the session account (`SenderCompID`).	String
SecurityID [48]	N	Optional filter. When set, restricts the response to one product; it is interpreted as a **`ProductId`** (not an `InstrumentId`).	String (parseable integer)

Field	Required	Descr	Values / Formatting
Symbol [55]	N	Fallback used to resolve the `ProductId` when `SecurityID` is absent.	String
PosReqType [724], **ClearingBusinessDate [715]**, **AccountType [581]**	N	Accepted for FIX compliance but **not parsed** by the gateway.	—

NewOrderSingle [D]

- A new order is followed by either an `ExecutionReport` [8] or a `BusinessMessageReject` [j].
- Accepted orders' `ExecType` [150] can be `0` (New / order status), `1` (Partial Fill), `2` (Fill), or `I` (Order Status; see ExecutionReport).
- Rejected orders' `ExecType` [150] is `8` (Rejected).
- Orders immediately cancelled (e.g. non-marketable IOC/FOK) receive an `ExecType` [150] of `4` (Canceled).

Field	Required	Descr	Values / Formatting
ClOrdID [11]	Y	Unique client order id.	Int64
SecurityID [48]	Y	The `InstrumentId`.	String (parseable integer)
Symbol [55]	N	Ignored for routing (informational only).	String
Side [54]	Y	Order side.	1 = BUY, 2 = SELL
TransactTime [60]	Y	Required by protocol. The server timestamps the order with its own UTC time.	UTC Timestamp
OrderQty [38]	Y	Order quantity.	Float
OrdType [40]	Y	Order type. Only Market and Limit may be submitted via FIX.	1 = MARKET, 2 = LIMIT
Price [44]	Y/N	Required for Limit orders; must be ≥ 0.	Float
TimeInForce [59]	N	Defaults to GTC. Day/opening/closing TIFs are not valid in a 24-hour market.	1 = GTC (default), 3 = IOC, 4 = FOK
Account [1]	N	Ignored — overridden by the session account.	String (parseable integer)
MaxShow [210]	N	Iceberg display quantity. Defaults to the full quantity when ≤ 0.	Integer
ExecInst [18]	N	Omit unless specifying Post-Only. Valid only on Limit orders.	6 = Post Only (Participate, don't initiate)
Text [58]	N	Ignored by the gateway.	String

OrderCancelRequest [F]

Cancels a single working order. The gateway keys the cancel solely on `OrigClOrdID` + `ClOrdID`; other fields are ignored.

Field	Required	Descr	Values / Formatting
OrigClOrdID [41]	Y	The `ClOrdID` of the order to cancel.	Int64
ClOrdID [11]	Y	A fresh unique id for this cancel request.	Int64
SecurityID [48]	N	Ignored.	—

Field	Required	Descr	Values / Formatting
Side [54]	N	Ignored.	—
Symbol [55]	N	Ignored (may be useful in your own logs).	—
TransactTime [60]	N	Ignored.	UTC Timestamp

OrderCancelReplaceRequest [G]

Replaces a working order (quantity/price/etc.). Internally the gateway submits a cancel-replace carrying the original order id plus a fresh order.

Field	Required	Descr	Values / Formatting
OrigClOrdID [41]	Y	`ClOrdID` of the order being replaced.	Int64
ClOrdID [11]	Y	A unique id for the replacement order.	Int64
SecurityID [48]	Y	The `InstrumentId`.	String (parseable integer)
Side [54]	Y	Order side.	1 = BUY, 2 = SELL
OrderQty [38]	Y	New quantity.	Float
OrdType [40]	Y	Order type (Market/Limit only).	1 = MARKET, 2 = LIMIT
Price [44]	Y/N	Required for Limit orders; must be ≥ 0.	Float
TimeInForce [59]	N	Defaults to GTC.	1 = GTC, 3 = IOC, 4 = FOK
MaxShow [210]	N	Iceberg display quantity.	Integer
ExecInst [18]	N	Post-Only (Limit only).	6 = Post Only
Account [1], Symbol [55], TransactTime [60]	N	Ignored.	—

Rejects arrive as an `OrderCancelReject` [9] with `CxlRejResponseTo = 2` (Cancel/Replace).

OrderStatusRequest [H]

Responds with a single `ExecutionReport` [8] (`ExecType = I`, Order Status), including `OrdStatusReqID` if provided. **Supply either `ClOrdID` or `OrderID`** — if neither is present, the gateway returns a `BusinessMessageReject` ("ClOrdID or OrderId must be supplied").

Field	Required	Descr	Values / Formatting
ClOrdID [11]	Y/N	Client order id of the order being queried (provide this **or** `OrderID`).	Int64
OrderID [37]	Y/N	Server order id of the order being queried.	Int64
OrdStatusReqID [790]	N	Optional correlator; echoed on the `ExecutionReport`.	String

> The order must be present in the gateway's per-session order shadow. Long-resting orders entered before this gateway instance started may not be found (`Failed to find OrderState`).

OrderMassStatusRequest [AF]

Streams one `ExecutionReport` [8] (`ExecType = I`) per order in the per-session shadow; the final report carries `LastRptRequested = Y` .

Field	Required	Descr	Values / Formatting
MassStatusReqID [584]	Y	Request id, echoed as `OrdStatusReqID` on each report.	String
MassStatusReqType [585]	Y	Only type **7** (Status for all orders) is supported; other values are rejected with a `BusinessMessageReject` .	7

> The shadow retains up to `MAXCOUNT = 5000` order states.

OrderMassCancelRequest [q]

Cancels working orders for the session's account. Responds with an `OrderMassCancelReport` [r] followed by an `ExecutionReport` [8] (`ExecType = 4` , Canceled) per successfully cancelled order.

Field	Required	Descr	Values / Formatting
ClOrdID [11]	Y	Request id.	String
MassCancelRequestType [530]	Y	**7** = cancel all orders; **1** = cancel all orders for one security.	7 or 1
SecurityID [48]	Y/N	Required when `MassCancelRequestType = 1` — the `InstrumentId` to cancel.	String (parseable integer)
TransactTime [60]	N	Timestamp of request.	UTC Timestamp

Unsupported request types return an `OrderMassCancelReport` with `MassCancelResponse = 0` (rejected) and a `MassCancelRejectReason` .

SecurityListRequest [x]

Field	Required	Descr	Values / Formatting
SecurityReqID [320]	Y	Request id, echoed on the response.	String
SecurityListRequestType [559]	Y	**4** (ALL_SECURITIES) returns the full instrument list including JSON metadata in `Text` [58]. Any other value returns `<Symbol, SecurityID>` pairs only.	4 = ALL_SECURITIES; other = symbols only

MarketDataRequest [V]

Level 2 book and trade-print subscriptions.

Field	Required	Descr	Values / Formatting
MDReqID [262]	Y	Unique identifier for the request; echoed on all responses and required to unsubscribe.	String

Field	Required	Descr	Values / Formatting
SubscriptionRequestType [263]	Y	Subscription action.	0 = Snapshot, 1 = Snapshot + Updates (Subscribe), 2 = Disable previous (Unsubscribe)
NoMDEntryTypes [267]	Y	Number of `MDEntryType` entries that follow.	Integer
MDEntryType [269]	Y	Type of market data requested (repeating group). Other values are rejected.	0 = Bid, 1 = Offer, 2 = Trade
NoRelatedSym [146]	Y	Number of instruments requested.	Integer
SecurityID [48]	Y	The `InstrumentId` (repeating group, per instrument).	String (parseable integer)
Symbol [55]	N	Ignored.	String
MarketDepth [264]	N	Accepted but ignored (full book only).	Integer
MDUpdateType [265]	N	Ignored — snapshot vs. incremental is implied by `SubscriptionRequestType`.	Integer

Subscription cap: a session may hold at most `MaxMDSubscriptionsPerSession` (default **100**) instrument subscriptions. **Over-cap requests are silently dropped** — the gateway logs a warning but does **not** send a `MarketDataRequestReject`. Track your own subscription count. (Rejects *are* sent for an unsupported `MDEntryType` or `SubscriptionRequestType`; see MarketDataRequestReject [Y].)

Server Messages (Gateway → Client)

ExecutionReport [8]

Used to confirm order receipt, report state changes, report fills, reject orders, and respond to status requests.

Field	Descr	Values / Formatting
OrderID [37]	Server-assigned unique order id.	Int64
ClOrdID [11]	Client-supplied order id.	String
OrigClOrdID [41]	Set when the report is for a replace.	String
ExecID [17]	Unique id of this execution message. `AssignedInputId` for state reports, `TradeId` for fills, `0` for status responses.	Int64
ExecType [150]	Purpose of this report. Note: fills reported via the order-state stream carry `I`; the dedicated fill path uses `1` / `2`. `F` (Trade) is not emitted.	0 = New, 1 = Partial Fill, 2 = Fill, 4 = Canceled, 8 = Rejected, C = Expired, I = Order Status
OrdStatus [39]	Current order state.	0 = New, 1 = Partially Filled, 2 = Filled, 4 = Canceled, 8 = Rejected, C = Expired
OrdRejReason [103]	Set when `OrdStatus = 8`. Values actually emitted: 1, 2, 11, 13, 18, 98, 99 (see Reject Reason Codes).	1 = Unknown Symbol, 2 = Exchange Closed, 11 = Unsupported Order Characteristic, 13 = Incorrect Quantity, 18 = Invalid Price Increment, 98 = Self-Trade Prevention, 99 = Other (see Text)
Account [1]	Account id.	String

Field	Descr	Values / Formatting
Symbol [55]	Instrument symbol.	String
SecurityID [48]	`InstrumentId`.	Integer
Side [54]	Order side.	1 = BUY, 2 = SELL
OrderQty [38]	Original order quantity.	Float
OrdType [40]	Order type. Reports may carry Stop / Stop-Limit / Pegged types for orders entered via other channels, even though those cannot be submitted over FIX.	1 = MARKET, 2 = LIMIT (3 = Stop, 4 = Stop-Limit, P = Pegged on reports)
Price [44]	Limit / stop-limit price.	Float
StopPx [99]	Stop trigger price (stop variants).	Float
PeggedPrice [839]	Peg offset for trailing-stop variants.	Float
TimeInForce [59]	Time in force.	1 = GTC, 3 = IOC, 4 = FOK
LastQty [32]	Quantity of this (last) fill. Set on fills.	Float
LastPx [31]	Price of this (last) fill. Set on fills.	Float
GrossTradeAmt [381]	`Price × Quantity` of this fill. Set on fills.	Float
LeavesQty [151]	Quantity still open. `OrderQty − CumQty`, or 0 when the order is no longer active.	Float
CumQty [14]	Cumulative filled quantity. (0 only on outright rejects.)	Float
AvgPx [6]	**Not populated — always 0.** Compute client-side from fills if needed.	Float (always 0)
MaxShow [210]	Iceberg display quantity, when set.	Integer
OrdStatusReqID [790]	Echoed from an `OrderStatusRequest`.	String
LastRptRequested [911]	Set on `OrderMassStatusRequest` responses (`Y` on the final report).	Boolean
TransactTime [60]	Time of the transaction represented by this report (OMS timestamp, UTC).	UTC Timestamp
Text [58]	Extra information when available (e.g. `Reject Code: <n>` for unmapped rejects).	String

> `UseOrderStatusExecType`: when set on the session (the shipped `Acceptor.cfg` default) or when responding to a status request, `ExecType` is forced to `I` (Order Status) regardless of the underlying transition. Many buy-side OMSs require this.

OrderCancelReject [9]

Field	Descr	Values / Formatting
OrderID [37]	Order id the cancel/replace was for.	Integer
ClOrdID [11]	`ClOrdID` of the cancel/replace request.	String

Field	Descr	Values / Formatting
OrigClOrdID [41]	`ClOrdID` of the order that could not be cancelled/replaced.	String
OrdStatus [39]	Always **4** (Canceled).	4
CxlRejResponseTo [434]	Which request this rejects.	1 = Order Cancel Request, 2 = Order Cancel/Replace
CxlRejReason [102]	Reason.	0 = Too Late To Cancel (order not found / already done), 99 = Other (reject code in Text)
Text [58]	`Reject Code: <n>` for the "Other" case.	String

OrderMassCancelReport [r]

Field	Required	Descr	Values / Formatting
ClOrdID [11]	Y	`ClOrdID` from the request.	String
OrderID [37]	Y	Unique id for this response.	String
MassCancelRequestType [530]	Y	Echoed request type.	7 = Cancel All Orders, 1 = Cancel Orders for a Security
MassCancelResponse [531]	Y	Result.	0 = Rejected, 7 = Cancel All Orders
MassCancelRejectReason [532]	N	Present when `MassCancelResponse = 0`.	0 = Mass Cancel Not Supported, 1 = Invalid or Unknown Security
TotalAffectedOrders [533]	N	Number of orders affected.	Integer
NoAffectedOrders [534]	N	Number of order identifiers that follow.	Integer
⇒ **OrigClOrdID [41]**	N	Repeated per affected order.	String

PositionReport [AP]

Sent once **per product** on the account. Tag numbers below are the standard FIX 4.4 tags emitted by the gateway.

Field	Required	Descr	Values / Formatting
PosMaintRptID [721]	Y	Per-process report counter.	String
PosReqID [710]	Y	Echoed from `RequestForPositions`.	String
PosReqResult [728]	Y	Result of the request.	0 = Valid request
Account [1]	Y	Account id.	String (parseable integer)
AccountType [581]	Y	1 = Customer side of books.	Integer
Symbol [55]	Y	Product symbol.	String
SecurityID [48]	Y	`ProductId` (not `InstrumentId`).	Integer
ClearingBusinessDate [715]	Y	UTC date of the report.	Date
NoPositions [702]	Y	Position group count (`1`).	Integer
⇒ **PosType [703]**	Y	Position type.	TQ = Transaction Quantity
⇒ **LongQty [704]**	Y	Amount (available balance).	Decimal

Field	Required	Descr	Values / Formatting
⇒ **ShortQty [705]**	Y	Hold (encumbered amount).	Decimal

> The gateway also sets `UnsolicitedIndicator (325)`, `SettlPrice (730)` / `SettlPriceType (731)` / `PriorSettlPrice (734)` (currently 0 / theoretical), and a `NoPosAmt (753)` group with `PosAmtType (707) = CASH` and `PosAmt (708) = Amount`.

MarketDataSnapshotFullRefresh [W]

Sent in response to `SubscriptionRequestType = 0` (Snapshot), one message per requested side.

Field	Required	Descr	Values / Formatting
MDReqID [262]	Y	Original `MDReqID` from the request.	String
NoMDEntries [268]	Y	Number of entries.	Integer
⇒ **MDUpdateAction [279]**	Y		0 = New, 1 = Change, 2 = Delete
⇒ **MDEntryType [269]**	Y		0 = Bid, 1 = Offer
⇒ **SecurityID [48]**	Y	`InstrumentId`.	Integer
⇒ **Symbol [55]**	Y		String
⇒ **MDEntryPx [270]**	Y	Price.	Float
⇒ **MDEntrySize [271]**	Y	Size.	Float
⇒ **MDEntryDate [272]**	Y		UTC Date
⇒ **MDEntryTime [273]**	Y		UTC Time
⇒ **NumberOfOrders [346]**	Y	Aggregate order count at the level.	Integer

MarketDataIncrementalRefresh [X]

Delivers the initial book (as `MDUpdateAction = 0` New per level) and subsequent deltas for `SubscriptionRequestType = 1`, plus trade prints.

Field	Required	Descr	Values / Formatting
MDReqID [262]	Y	Original `MDReqID`.	String
NoMDEntries [268]	Y	Number of entries.	Integer
⇒ **MDUpdateAction [279]**	Y		0 = New, 1 = Change, 2 = Delete
⇒ **MDEntryType [269]**	Y		0 = Bid, 1 = Offer, 2 = Trade
⇒ **SecurityID [48]**	Y	`InstrumentId`.	Integer
⇒ **Symbol [55]**	Y		String
⇒ **MDEntryPx [270]**	Y	Price.	Float
⇒ **MDEntrySize [271]**	Y	Size.	Float
⇒ **MDEntryDate [272]**	Book entries	Set for Bid/Offer entries.	UTC Date

Field	Required	Descr	Values / Formatting
⇒ **MDEntryTime [273]**	Book entries	Set for Bid/Offer entries.	UTC Time
⇒ **NumberOfOrders [346]**	Book entries	Aggregate order count (Bid/Offer entries).	Integer
⇒ **TickDirection [274]**	Trade entries	Set for Trade entries only.	0 = Up (Plus), 1 = Zero-minus, 2 = Down (Minus)

MarketDataRequestReject [Y]

Sent for an unsupported `MDEntryType` or `SubscriptionRequestType`. (It is **not** sent for over-cap subscriptions — those are dropped silently.)

Field	Required	Descr	Values / Formatting
MDReqID [262]	Y	Echoed from the request.	String
MDReqRejReason [281]	Y	Reason.	D = Unsupported MDEntryType, E = Unsupported SubscriptionRequestType
Text [58]	N	Human-readable reason.	String

SecurityList [y]

Field	Required	Descr	Values / Formatting
SecurityReqID [320]	Y	Echoed from the request.	String
SecurityResponseID [322]	Y	Unique response id (GUID).	String
SecurityRequestResult [560]	Y		0 = Valid Request
NoRelatedSym [146]	Y	Number of securities returned.	Integer
⇒ **Symbol [55]**	N	Instrument symbol.	String
⇒ **SecurityID [48]**	Y	`InstrumentId`.	Integer
⇒ **Text [58]**	N	JSON representation of the AP instrument definition — populated only for `SecurityListRequestType = 4`.	String (JSON)

BusinessMessageReject [j]

Application-layer reject for parse errors, rate limits, and OMS-unavailable conditions.

Field	Required	Descr	Values / Formatting
RefSeqNum [45]	N	`MsgSeqNum` of the rejected message.	Integer
RefMsgType [372]	Y	`MsgType` of the referenced message.	String

Field	Required	Descr	Values / Formatting
BusinessRejectReason [380]	Y	Reason. Values actually emitted: 0, 2, 18.	0 = Other (parse error, server error, OMS unavailable, rate-limited), 2 = Unknown Security, 18 = Invalid Price Increment
Text [58]	N	Human-readable reason, e.g. `OMS unavailable`, `Rate Limit`, `Invalid account`, `SecurityID is required`, `Invalid Order Type`, `Invalid Side`, `TimeInForce only Good_Till_Cancel, Fill_Or_Kill, and Immediate_Or_Cancel are supported`, `PostOnly can only be used with a Limit Order.`	String

Reject Reason Codes

These OMS reject codes may appear in `Text` [58] (formatted `Reject Code: <n>`) on rejects that map to `OrdRejReason = 99` / `CxlRejReason = 99`. Codes that map to a specific FIX reason instead populate `OrdRejReason` [103] or `CxlRejReason` [102] directly (see the mapping below).

```
10001 = Invalid Instrument (Invalid Product Pair)
10008 = Invalid Remaining Quantity
10009 = Invalid Stop Price
10010 = Invalid Limit Price
10011 = Invalid Client OrderId
10018 = Invalid Display Quantity
10200 = Unknown Instrument (Unknown Product Pair)
10207 = Unknown Side
10208 = Unknown Order Type
10209 = Unknown Peg Price Type
10300 = Unsupported Peg Price Type
10500 = No Buy Side Market
10501 = No Sell Side Market
10502 = Not Enough Funds
10509 = Order Not Found
10512 = Edit Not Supported
10513 = Market Order Entered with Limit Price
10515 = Remaining Quantity Below Minimum
10516 = Display Quantity Below Minimum
10517 = Exchange Not Found
10518 = Client OrderId Not Found
10520 = Self Trade Prevention
10524 = Invalid Limit Price Below Floor
10525 = Invalid Limit Price Above Ceiling
10526 = Invalid Stop Price Below Floor
10527 = Invalid Stop Price Above Ceiling
11000 = Market Closed
11001 = Market Paused
```

OrdRejReason mapping (`OrdStatus = 8`)

OMS code(s)	FIX `OrdRejReason` [103]
11000, 11001	2 = Exchange Closed
10001, 10200	1 = Unknown Symbol
10008	13 = Incorrect Quantity
10010	18 = Invalid Price Increment
10207, 10208	11 = Unsupported Order Characteristic
10301	11 + `Text = "Order Type Unsupported"`
10520 (self-trade)	98 = Self-Trade Prevention (custom)
any other	99 = Other + `Text = "Reject Code: <n>"`

CxlRejReason mapping

OMS code(s)	FIX `CxlRejReason [102]`
10509, 10518	0 = Too Late To Cancel
any other	99 = Other + `Text = "Reject Code: <n>"`

Connection Lifecycle

```
1. TCP connect ──► gateway accepts the socket
2. Send Logon (A):
     SenderCompID = <your AccountId>, TargetCompID = <ServerName>,
     Username (553) + Password (554)
3. Gateway runs:
     - MaxSessions cap check
     - (per-IP logon rate check — currently inert)
     - VerifyUser: credentials, locked, email-verified, 2FA-disabled,
       account membership (FIX-enabled)
4a. ✓ Authenticated ──► gateway responds with Logon (A); session is live
4b. ✗ Failed        ──► gateway responds with Logout (5)
       (text: "Logon failed" / "Max sessions reached" / "Logon rate limit exceeded")
5. Application traffic flows
6. Logout (5) initiated by either side
     ↳ If the session has CancelOnDisconnect=Y (the shipped default),
       the gateway cancels all working orders for the account first.
```

Order-Entry Walkthrough

Submit a Limit order

`NewOrderSingle (D)` — e.g. buy 0.5 of instrument 27 at 68500, GTC:

```
8=FIX.4.4 | 35=D | 49=48201 | 56=APEXFIX-PROD | 34=2 | 52=... |
11=1000001 | 48=27 | 54=1 | 60=... | 38=0.5 | 40=2 | 44=68500.00 | 59=1 | 10=...
```

- `ClOrdID (11)` — must be a 64-bit integer you assign
- `SecurityID (48)` — APEX `InstrumentId`
- `Side (54) = 1` (Buy), `OrderQty (38) = 0.5`, `OrdType (40) = 2` (Limit), `Price (44) = 68500.00`, `TimeInForce (59) = 1` (GTC)

Acknowledgement

Gateway → client: `ExecutionReport (8)` with `OrdStatus = 0` (New) and `ExecType = I` (when `UseOrderStatusExecType = Y`, the default) or `0` (New). The OMS-assigned `OrderID (37)` is now bound to your `ClOrdID`.

Fill

For each match: `ExecutionReport (8)` with `ExecType = 1` (Partial Fill) / `2` (Fill), `OrdStatus = 1` / `2`, and `LastPx (31)`, `LastQty (32)`, `GrossTradeAmt (381)`, `ExecID (17) = TradeId`. Track `LeavesQty (151)` — when it reaches 0 the order is done.

Cancel

`OrderCancelRequest (F)`:

```
8=FIX.4.4 | 35=F | 49=48201 | 56=APEXFIX-PROD | 11=2000001 | 41=1000001 | 10=...
```

- `ClOrdID (11)` — a fresh id for the cancel request itself
- `OrigClOrdID (41)` — the `ClOrdID` of the order to cancel

Outcomes: success → `ExecutionReport (8)` `ExecType = 4` (Canceled); failure → `OrderCancelReject (9)` with `CxlRejReason` and `Text`.

Replace / Mass Cancel / Status

- `OrderCancelReplaceRequest (G)` — replace an order (Market/Limit only).
- `OrderMassCancelRequest (q)` — `MassCancelRequestType = 7` cancels every working order; `= 1` plus `SecurityID (48)` limits to one instrument.
- `OrderStatusRequest (H)` — supply `ClOrdID` or `OrderID`.
- `OrderMassStatusRequest (AF)` with `MassStatusReqType = 7` — enumerate the per-session shadow.

Recommended Client Patterns

- **`ClOrdID` is int64.** Use a monotonic counter (`(epochSecond << 24) | sequence` works well). Don't reuse a `ClOrdID` within a trading day for the same account.
- **Handle** `BusinessMessageReject (j)` — it carries parse errors, rate limits, and OMS-unavailable conditions.
- On `Text = "OMS unavailable"`, **back off** — retrying immediately stays rejected. Pause before retrying.
- **Honor the rate limits.** The overall per-session bucket refills at `RateLimit`/sec; on `Text = "Rate Limit"`, throttle. Don't reconnect-storm.
- **After reconnect** (sequence numbers reset via `ResetOnLogon=Y`): issue `OrderMassStatusRequest (AF)` (`MassStatusReqType = 7`) and reconcile against your local book by `ClOrdID` / `OrderID`. If the previous session had `CancelOnDisconnect=Y`, expect the working-order list to be empty.
- **Compute `AvgPx` yourself** from fills — the gateway always reports `AvgPx = 0`.

- **Use TLS** for any non-trivial deployment (`UseSSL=true` on the acceptor; standard PFX certificate).

End-to-End Smoke Test

1. Connect TCP, send `Logon (A)` with valid credentials → expect `Logon (A)` back.
2. `SecurityListRequest (x)` with `SecurityListRequestType = 4` → expect `SecurityList (y)`; note the `SecurityID` for your test instrument.
3. `MarketDataRequest (V)` with `SubscriptionRequestType = 0`, bid (`269=0`) and offer (`269=1`) → expect two `MarketDataSnapshotFullRefresh (W)`.
4. `NewOrderSingle (D)` for a small Limit order well outside the book → expect `ExecutionReport (8)` with `OrdStatus = 0` (New).
5. `OrderStatusRequest (H)` with the `ClOrdID` → expect `ExecutionReport (8)` with `ExecType = I`.
6. `OrderCancelRequest (F)` → expect `ExecutionReport (8)` with `OrdStatus = 4` (Canceled).
7. `Logout (5)` → expect `Logout (5)` back; socket closes.

Example Initiator Config

A minimal QuickFIX initiator `.cfg` to connect to the gateway:

```
#QuickFix - default settings for sessions
[DEFAULT]
FileStorePath=store
FileLogPath=log
ConnectionType=initiator
ReconnectInterval=60

# session definition
[SESSION]
BeginString=FIX.4.4
TargetCompID=APSender          # the gateway's ServerName, provided by AP staff
SenderCompID=MyAccountID       # your APEX Account ID (integer)
HeartBtInt=30
SocketConnectPort=9821
SocketConnectHost=127.0.0.1
DataDictionary=spec/FIX44.xml
ResetOnLogon=Y
StartTime=00:00:00
EndTime=23:59:59
```

Supply `Username (553)` and `Password (554)` on the `Logon (A)` message (in most QuickFIX implementations, via the `ToAdmin` callback).