FIX protocol

Protocol documentation

> *Architect:* *Aleksandr Baryshnikov*

Revision History

Version	Date	Note
1.0	February 05, 2019	First version

Introduction

This document describes the set and formats of blocks, fields and messages of FIX 4.4 Protocol (to find reasonable information on onixs.biz or fixtrading.org.), adapted for information exchange with the electronic trading system of the market of the CRIX.IO (hereinafter – CRIX). The trading system is connected through an intermediate FIX Gateway (hereinafter —gateway).

Protocol adaptation means that fields may be added or deleted, as well as requirement attributes of fields in a message or group may be changed in messages and groups described by FIX 4.4 Protocol Specification. The adaptation also extends the set of messages. For fields accepting a defined set of values this set may be added. During the message analysis, fields failed to be described in this document will be ignored.

Audience

The documents may be useful to whom plans to integrate with CRIX.IO exchange over FIX protocol. Assumes that basic technical related things like FIX protocol or TCP already familiar for a reader, however it's possible to find reasonable information on onixs.biz or fixtrading.org.

Connection

	Development	MVP	Production
Host	fix.dev.crix.io	fix.mvp.crix.io	fix.crix.io
Port	9823	9823	9823
SSL	no	no	no

- Fix protocol: 4.4
- All sequence are reset after logon.
- **SenderCompID** should be a API token issued as a part of BOT API
- The Logon (MsgType=A) message should be sent after connection and before first operation

Example config

```
[DEFAULT]
FileLogPath=logs
ReconnectInterval=60
HeartBtInt=20


[SESSION]
BeginString=FIX.4.4
TargetCompID=CRIX
SenderCompID=vcggnzu1hp1itbiawf7v4rsav68h1z0mdvv9hqk9034dn6l74lt6ochr8gzev05i
ConnectionType=initiator
SocketConnectPort=9823
SocketConnectHost=dev.crix.io
DataDictionary=dicts/FIX44.xml
FileStorePath=store
```

Messages

Only required fields are listed here. Header and trailer should also be filled as it's described in a protocol.

Logon

It is used by the client to initiate a session start. Each session should start with sending of this message. The gateway uses it to confirm the start of session.

MsgType =A

Field	Type	Description
Password(554)	text	Secret key as provided by the BOT API

NewOrderSingle

MsgType=D
This message is used to send a new order to the trading system.

(Create new order if possible)

Field	Type	Description
Side (54)	char	Buy (1) or Sell (2)
Symbol(55)	text	Symbol name as-is (for example: ETH_BTC)
OrdType(40)	char	Market(1), Limit (2)
ClOrdId(11)	text	Any text, ignored by gateway but required by protocol. Possibly will be implemented in a future
TimeInForce(58)	char	Specifies how long the order remains in effect • 1 = Good Till Cancel (GTC) • 3 = Immediate or Cancel (IOC) • 4 = Fill or Kill (FOK)
StopPx(99)	double	(optional) stop price for Stop Loss/Take profit orders
Price(44)	double	Price
OrderQty(38)	double	Quantity
TransactTime(60)	timestamp	Ignored by gateway but required by protocol

Reports

After successful connection of the client and session starting, the client automatically starts receiving Execution Report messages.

MsgType=8

SUCCESS

Field	Type	Description
Symbol(55)	text	Symbol name as-is (for example: ETH_BTC)
Side (54)	char	Buy (1) or Sell (2)
ExecID (17)	text	Message ID (should be unique per session)
ExecType(150)	char	New(0)

OrdStatus(39)	char	New(0)
LeavesQty(151)	double	Quantity, same as in request
CumQty(14)	double	0
AvgPx(6)	double	0
ClOrdId(11)	text	Same text in request if presented. Can be used for correlation

Reject

It is used by both parties in case of loss of transmission sequence of session-level messages, and in case of violation of the message format.

Field	Type	Description
Symbol(55)	text	Symbol name as-is (for example: ETH_BTC)
Side (54)	char	Buy (1) or Sell (2)
ExecID (17)	text	Message ID (should be unique per session)
ExecType(150)	char	Rejected (8)
Text(58)	text	Reject description
OrdRejReason(103)	text	Other(99)
ClOrdId(11)	text	Same text in request if presented. Can be used for correlation

OrderMassStatusRequest

MsgType=AF

Get status of all active orders

Field	Type	Description
Symbol(55)	text	Symbol name as-is (for example: ETH_BTC)
MassStatusReqID(584)	text	Request ID for reply correlation
MassStatusReqType(585)	char	Status for orders for a security (1)

Reports

(execution reports, MsgType=8)

SUCCESS

Field	Type	Description
Symbol(55)	text	Symbol name as-is (for example: ETH_BTC)
Side (54)	char	Buy (1) or Sell (2)
ExecID (17)	text	Message ID (should be unique per session)
TotNumReports(911)	int	Total number of open orders
ExecType(150)	char	OrderStatus(I (capital i))

MassStatusReqID(584)	text	Request ID for reply correlation
WorkingIndicator(636)	bool	Order Is Currently Being Worked (true)
OrdStatus(39)	char	New(0)
LeavesQty(151)	double	Quantity, same as in request
CumQty(14)	double	0
AvgPx(6)	double	0
Account(1)	text	User ID
OrderID(37)	text	ID of order given by exchange

OrderCancelRequest

MsgType=F

Cancel single order

Field	Type	Description
Symbol(55)	text	Symbol name as-is (for example: ETH_BTC)
OrderID(37)	text	ID of order given by exchange
Side (54)	char	Buy (1) or Sell (2)
ClOrdId(11)	text	Ignored by gateway but required by protocol.
OrigClOrdID(41)	text	Ignored by gateway but required by protocol
TransactTime(60)	timestamp	Ignored by gateway but required by protocol

Reports

(execution reports, MsgType=8)

SUCCESS

Field	Type	Description
Symbol(55)	text	Symbol name as-is (for example: ETH_BTC)
Side (54)	char	Buy (1) or Sell (2)
ExecID (17)	text	Message ID (should be unique per session)
ExecType(150)	char	Canceled (4)
OrdStatus(39)	char	Canceled (4)
LeavesQty(151)	double	Initial quantity - Filled quantity
CumQty(14)	double	Filled quantity
AvgPx(6)	double	0
OrderQty(38)	double	Initial order quantity
ClOrdId(11)	text	Same text in request if presented. Can be used for correlation
Account(1)	text	User ID

OrderID(37)	text	ID of order given by exchange

REJECT

Field	Type	Description
Symbol(55)	text	Symbol name as-is (for example: ETH_BTC)
ExecID (17)	text	Message ID (should be unique per session)
ExecType(150)	char	Rejected (8)
Text(58)	text	Reject description
OrdRejReason(103)	text	Other(99)
ClOrdId(11)	text	Same text in request if presented. Can be used for correlation

TradeCapture

MsgType=AE

Sends by exchange for each trade related to user orders

Field	Type	Description
TradeReportID(571)	text	Unique trade ID
ExecID (17)	text	Message ID (should be unique per session)
TrdType(828)	int	Regular(0)
ExecType(150)	char	Trade (F)
Symbol(55)	text	Symbol name as-is (for example: ETH_BTC)
QtyType(854)	int	Units(0)
LastQty(32)	double	Trade quantity
LastPx(31)	double	Trade price
PreviouslyReported(570)	bool	Always false
TradeDate(75)	timestamp (yyyyMMdd)	Trade date
TransactTime(60)	timestamp	Trade time
NoSides(552)	int	2 (two sides: buy and sell)
Side 1 (buy)		
Side (54)	char	Buy (1)
OrdStatus(39)	char	New(0) or Filled(2) if complete
OrderID(37)	text	ID of order given by exchange
Side 2 (sell)		
Side (54)	char	Sell (0)
OrdStatus(39)	char	New(0) or Filled(2) if complete
OrderID(37)	text	ID of order given by exchange

MarketDataRequest

MsgType=V

Subscribe to market data (changes/changes + snapshot)

Field	Type	Description
`MDReqID(262)`	text	Any value, required by protocol, ignored by system
`SubscriptionRequestType(263)`	char	0 = Snapshot 1 = Snapshot + Updates (Subscribe) 2 = Disable previous Snapshot + Update Request (Unsubscribe)
`MarketDepth(264)`	char	Only Top Of Book supported (1)
`NoMDEntryTypes(267)`	int	2 (bids and offers)
bids (NoMDEntryTypes group)		
`>>> MDEntryType(269)`	char	0 = Bid
offers (NoMDEntryTypes group)		
>>> MDEntryType(269)	char	1 = Offer
`NoRelatedSym(146)`	int	Number of symbols you want to subscribe
each symbol (NoRelatedSym group)		
>>> Symbol(55)	text	Symbol name to subscribe or unsubscribe

Market Data - Snapshot/Full Refresh

MsgType=W

Market data update (depth - top of book)

Field	Type	Description
`Symbol(55)`	text	Symbol name
`NoMDEntries(268)`	int	Number of entries (sum of num asks + num bids on top of order book)
`>> MDEntryType(269)`	char	0 = Bid, 1 = Offer
`>> MDEntryPx(270)`	double	entry price
`>> MDEntrySize(271)`	int	Number of orders
>> MinQty(110)	double	Volume of orders