

Wincent OTC FIX API

This document contains the FIX interface specification for Wincent OTC API. The document is intended to be used by clients as a technical reference when building interactions with Wincent OTC through FIX.

Supported features:

1. Subscribe to streaming prices
2. Send fill or kill (FOK) limit orders at specific prices
3. Send immediate or cancel (IOC) limit orders at specified prices that can be partially filled

NOTE

The streamed prices are only indicative. Interpolating pricing based off of existing pricing levels may lead to an increase in rejections, as our pricing structure isn't linear.

The Wincent OTC FIX API supports a subset of the FIX 4.4 protocol.

Connection details

The Wincent OTC API supports two session types:

- Market-Data session - for subscribing to streaming prices
- Trading session - for submitting orders and receiving fills

The services are hosted in AWS. Access to the platform is supported through TCP/SSL encrypted TCP connection. Both UAT and Production source IP addresses need to be whitelisted separately. Clients are not required to complete a conformance test, however failure to comply to Wincent's OTC FIX specification may lead to disabling the session, without notice if necessary.

The connection details will be shared through direct channels.

Authentication

For security reasons, the Wincent FIX API uses:

- `API_KEY`
- `API_SECRET_KEY`

The `Logon <A>` message must be signed by the client. The `Username <553>` should be set to the `API_KEY`. The `RawData <96>` should contain the current timestamp (strictly increasing integer) and *nonce* (base64-encoded randomly chosen bytes) separated by ASCII period (.). The

Password <554> field should contain the base64 encoded SHA384 HMAC signature of RawData <96> using the `API_SECRET_KEY`.

Some FIX engines might have issues with base64 padding character `=`, so the nonce should be of length divisible by 3 (e.g. 48). Similarly the signatures with SHA384 should not include any `=` padding characters.

▼ NodeJS example

```
const crypto = require('crypto')

const API_KEY = 'dce7a604-787e-497b-a1b4-42d214551426'
const SECRET =
Buffer.from('8a2cc0673b1c428315fe84c0138d95c3ddda30baf81e7d9aa821f1ca47098
193', 'base64')

const nonce = crypto.randomBytes(48).toString('base64')
const rawData = `${Date.now().toFixed(0)}.${nonce}`
//
1703231636860.aEYxzBFdCkh/4+nKTRR7vxtXHNjcQpoZzzpYk9qWF9r1nFiCl0Y3//I4t9LD
JcqB

const hmac = crypto.createHmac('sha384', SECRET)
const data = hmac.update(rawData)
const signature = data.digest('base64')
// 1EjZwGXaGqV+mGg3si/8lSYNkRX4FhiIhFxV+1u8i71pgaYj8m8c6gZUcpuajXXW

const logon_payload = [
  //... header
  `98=0`, // EncryptMethod = 0 None
  `108=30`, // HeartBInt - 30s
  `95=${rawData.length}`, // RawDatLength
  `96=${rawData}`, // RawData
  `141=Y`, // ResetSeqNumFlag
  `553=${API_KEY}`, // Username
  `554=${signature}`, // Password
  //... trailer
].join('\u0001')
//
98=0\x01108=30\x0195=78\x0196=1703231636860.aEYxzBFdCkh/4+nKTRR7vxtXHNjcQp
oZzzpYk9qWF9r1nFiCl0Y3//I4t9LDJcqB\x01141=Y\x01553=dce7a604-787e-497b-
a1b4-
42d214551426\x01554=1EjZwGXaGqV+mGg3si/8lSYNkRX4FhiIhFxV+1u8i71pgaYj8m8c6g
ZUcpuajXXW
```

▼ Python example

```

import time
import secrets
import base64
import hmac
import hashlib

API_KEY = 'dce7a604-787e-497b-a1b4-42d214551426'
SECRET =
'8a2cc0673b1c428315fe84c0138d95c3ddda30baf81e7d9aa821f1ca47098193'

nonce = base64.b64encode(secrets.token_bytes(48)).decode('utf-8')
raw_data = '{}.{}'.format(int(time.time()*1000), nonce)
#
1703231636860.aEYxzBFdCkh/4+nKTRR7vxtXHNjcQpoZzzpYk9qWF9r1nFiCl0Y3//I4t9LD
JcqB

signature = base64.b64encode(hmac.new(
    base64.b64decode(SECRET),
    msg=bytes(raw_data, 'utf-8'),
    digestmod=hashlib.sha384
).digest()).decode('utf-8')
# 1EjZwGXaGqV+mGg3si/8lSYNkRX4FhiIhFxV+1u8i71pgaYj8m8c6gZUcpuajXXW

logon_payload = '\x01'.join([
    # ... header
    '98=0',
    '108=30',
    f'95={len(raw_data)}',
    f'96={raw_data}',
    '141=y',
    f'553={API_KEY}',
    f'554={signature}'
    # ... trailer
])
#
98=0\x01108=30\x0195=78\x0196=1703231636860.aEYxzBFdCkh/4+nKTRR7vxtXHNjcQp
oZzzpYk9qWF9r1nFiCl0Y3//I4t9LDJcqB\x01141=Y\x01553=dce7a604-787e-497b-
a1b4-
42d214551426\x01554=1EjZwGXaGqV+mGg3si/8lSYNkRX4FhiIhFxV+1u8i71pgaYj8m8c6g
ZUcpuajXXW

```

For security reasons, please do not share your `API_KEY` or `API_SECRET_KEY` via any means, even with Wincent. Wincent staff will never request these keys from you.

FIX Session

Wincent does not persist FIX messages for session recovery between connections neither on trading or market data sessions. The sequence numbers should be reset for every new connection through the ResetSeqNumFlag <141> set to Y .

Quotes from a previous session are considered invalid. The current supported Orders (MKT, IOC and FOK) cannot be cancelled. Even if the connection is dropped between sending the orders and receiving the Execution Report <8>, the order may be processed by the API.

Certification Process

Before going to Production environment, the integration needs to be tested and verified on the UAT environment by Wincent. The Production environment credentials will be provided afterwards.

Clients are not required to complete a full conformance test, however failure to comply to the FIX specification may lead to disabling of the session, without notice if necessary.

Quoting and Trading Conventions

In all application messages, the symbol field defines the currency pair and instrument type.

The symbol format is <BASE>_<QUOTE>_<TYPE> where:

- BASE is the base currency
- QUOTE is the quote currency
- TYPE currently can be SPOT

For example: BTC_USDC_SPOT is for spot market Bitcoin against USDC stable coin.

All quantity fields will be in terms of the base currency.

Rates and prices are expressed as units of the quote currency per unit of the base currency. E.g. an order to 10 BTC_USDC_SPOT at 34500 means buying 10 BTC with \$345,000

Rate and price precision are expected at most 8 digits after the decimal point.

Standard header and trailer

Every message, whether administrative or application, is preceded by a StandardHeader. The header identifies the message type, length, destination, sequence number, origination point and time.

Tag	Required	Description
<u>BeginString <8></u>	Y	Identifies the beginning of a new message and protocol version. Valid value: FIX.4.4
<u>BodyLength <9></u>	Y	Length of message body not include the header/trailer
<u>MsgType <35></u>	Y	Defines the message type
<u>SenderCompID <49></u>	Y	Used to identify the counterparty sending the message and will be agreed upon during the onboarding process
<u>TargetCompID <56></u>	Y	Should be set to WINCENT_MD , WINCENT_TRADE (for messages sent by the client) based on the session type
<u>MsgSeqNum <34></u>	Y	Integer message sequence number
<u>SendingTime <52></u>	Y	UTC Timestamp of when the current message was transmitted. E.g: 20250430-21:43:58.419

All messages should include the following trailer:

Tag	Required	Description
<u>Checksum <10></u>	Y	Three bytes, simple checksum calculated for the entire message (including all header fields and body). Always last field in the message.

Admin messages

Logon <A>

Logon request should be sent by the client to start a FIX session. It must be the first message after establishing the connection. Only one session can be established per connection and any old connections will be dropped.

In the event of successful authorization the client receives a Logon auth confirmation.

In case of authorization error, Logout <5> message will be sent to the client, with Text <58> field of the following format: Auth error: <ERROR_CODE> , where ERROR_CODE is Wincent OTC API's error code.

For more details on the Username <553> and Password <554> details see Authentication.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Logon <A></u>

Tag	Required	Description
<u>EncryptMethod <98></u>	Y	Valid value: 0 (= None)
<u>HeartBlnt <108></u>	Y	Heartbeat interval in seconds. E.g. 30
<u>RawDataLength <95></u>	N	Length of the raw data. Not required as the normal <u>RawData <96></u> is a base64 encoded text here
<u>RawData <96></u>	Y	The timestamp and base64 encoded nonce. E.g. 1696340223549.YWJjZGVm
<u>ResetSeqNumFlag <141></u>	Y	Valid value: Y (= Reset)
<u>Username <553></u>	Y	API_KEY
<u>Password <554></u>	Y	SHA384 HMAC signature of <u>RawData <96></u> using the API_SECRET_KEY
Standard Trailer	Y	

The auth logon confirmation fields:

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Logon <A></u>
<u>EncryptMethod <98></u>	Y	Valid value: 0 (= None)
<u>HeartBlnt <108></u>	Y	Confirmed heartbeat interval
Standard Trailer	Y	

Logout <5>

Logout request sent by either side to terminate the session. The other side should respond with another Logout message to acknowledge session termination. The connection will be closed afterwards. All orders delivered before logout given they cannot be cancelled (MKT, IOC and FOK) can still be filled.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Logout <5></u>
<u>Text <58></u>	N	The reason for logging out
Standard Trailer	Y	

Test Request <1>

Test request can be used by either side to force sending Heartbeat <0> message. The recipient should reply with a Heartbeat <0> with TestReqId <146> set to the corresponding value

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Test Request <1></u>
<u>TestReqId <146></u>	Y	For example a timestamp.
Standard Trailer	Y	

Heartbeat <0>

Sent by either side if a message has not been received in the past HeartBInt <108> seconds. Should also be sent in response to a Test Request <1>.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Heartbeat <0></u>
<u>TestReqId <146></u>	C	Required if it's a response to a <u>Test Request <1></u> set to the corresponding <u>TestReqId <146></u> from the request.
Standard Trailer	Y	

Reject <3>

Sent as a response if a message is corrupted and doesn't fulfill FIX message format

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Reject <3></u>
<u>RefSeqNum <36></u>	Y	<u>MsgSeqNum <34></u> of the rejected message
<u>Text <58></u>	N	Optionally additional details for the rejection
Standard Trailer	Y	

Business Message Reject <j>

Sent as a response if a message is a valid FIX message, but the message does not comply with this specification.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Business Message Reject <j></u>
<u>RefSeqNum <36></u>	Y	<u>MsgSeqNum <34></u> of the rejected message

Tag	Required	Description
<u>RefMsgType <372></u>	Y	<u>MsgType <35></u> of the rejected message
<u>BusinessRejectReason <380></u>	Y	Valid values: 0 (= Other), 2 (= Unknown security), 3 (= Unsupported Message Type), 5 (= Conditionally required missing field)
<u>Text <58></u>	N	Optionally additional details for the rejection
Standard Trailer	Y	

Market Data

Security List Request <x>

The message can be used to request a list of securities the client has permission to trade. The server will respond with one or more Security List <y>.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Security List Request <x></u>
<u>SecurityReqID <320></u>	Y	Unique identifier for the market data request (e.g. UUIDv4)
<u>SecurityListRequestType <559></u>	Y	Valid value 0 (= Symbol)
Standard Trailer	Y	

Security List <y>

This message is sent as a response to Security List Request <x> and contains all securities that the client has permissions to trade. The whole list might be chunked into multiple messages.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Market Data Snapshot/Full Refresh <W></u>
<u>SecurityReqID <320></u>	Y	The unique identifier for the security list request that this message is in response to
<u>SecurityResponseID <322></u>	Y	Unique identifier for this response
<u>SecurityRequestResult <560></u>	Y	Result of the security request. Possible values: 0 (= Valid), 1 (= Invalid)
<u>TotNoRelatedSym <393></u>	N	The total number of securities if the message is fragmented.
<u>LastFragment <893></u>	N	Indicates if this is the last message if the response is fragmented. Values: Y (= last message), N (= not last message)

Tag	Required	Description
<u>NoRelatedSym <146></u>	Y	The number of traded symbols in this message
⇒ <u>Symbol <55></u>	Y	Symbol, e.g. BTC_USD_SPOT is a response to a specific quote
<u>Text <58></u>	N	Comment, instruction or error message if the request was invalid
Standard Trailer	Y	

Market Data Request <V>

The message is used to subscribe the current session to a stream of Market Data Snapshot/Full Refresh <W> messages. To avoid rejections it's recommended not to subscribe to all symbols in a single request.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Market Data Request <V></u>
<u>MDReqId <262></u>	Y	Unique identifier for the market data request (e.g. UUIDv4)
<u>SubscriptionRequestType <263></u>	Y	Either 1 (= Subscribe) or 2 (= Unsubscribe)
<u>MarketDepth <264></u>	Y	Valid value 0 (= Full Book)
<u>MDUpdateType <265></u>	Y	Valid value 0 (= Full Refresh)
<u>NoMDEntryTypes <267></u>	Y	The number of <u>MDEntryType <269></u> fields requested
⇒ <u>MDEntryType <269></u>	Y	The type of market data entry to receive snapshots and updates for. Value value: 0 (= Bid), 1 (= Offer)
<u>NoRelatedSym <146></u>	Y	The number of symbols requested
⇒ <u>Symbol <55></u>	Y	The symbol to get data for
Standard Trailer	Y	

Market Data Request Reject <Y>

The message is used when Wincent cannot honor the Market Data Request <V> due to business or technical reasons. For example, if the client does not have permission to trade one or more symbols listed.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Market Data Request Reject <Y></u>
<u>MDReqId <262></u>	Y	The unique identifier for the market data request that this message is in response to

Tag	Required	Description
<u>MDReqRejReason <281></u>	N	Rejection reason. Valid values see below.
<u>Text <58></u>	N	Contains Wincent Error code
Standard Trailer	Y	

Rejection reason code	Description
0	Unknown symbol
3	Insufficient permissions
W	Wincent-specific error. See <u>Text <58></u>

Market Data Snapshot/Full Refresh <W>

The message is used to transmit updates to the indicative prices. The update frequency might vary based on client requirements.

The quotes inside a single update should be considered NON-sweepable. Sweepable in this context meaning any number of quotes from a single snapshot being used to create orders. A simple example of a sweepable quote snapshot would be taking all quotes on one side up to some quantity level. This is in contrast with a non-sweepable quote distribution, where taker should use a quantity level that's higher than the total sum.

NOTE

NoMDEntries <268> might be absent or set to zero to signify an upstream pricing issue. This invalidates the most recent refresh where prices and quantities are no longer valid.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Market Data Snapshot/Full Refresh <W></u>
<u>MDReqId <262></u>	Y	The unique identifier for the market data request that this message is in response to
<u>Symbol <55></u>	Y	Symbol of the market data entry
<u>NoMDEntries <268></u>	Y	The number of entries following
⇒ <u>MDEntryType <269></u>	Y	The type of market data update. Valid values: 0 (= Bid), 1 (= Offer)
⇒ <u>MDEntryPx <270></u>	Y	Price of the market data entry
⇒ <u>MDEntrySize <271></u>	Y	Quantity of the market data entry

Tag	Required	Description
⇒ <u>QuoteEntryID <299></u>	Y	A unique identifier for the quote that should be sent in <u>New Order Single <D></u> in <u>QuoteID <117></u> if the order is a response to a specific quote
Standard Trailer	Y	

Trade

Trades can be executed via three different methods:

Limit Fill or Kill Order (FOK): an order which is sent with the price that the client wishes to trade at. It is executed immediately or rejected.

Limit Immediate or Cancel (IOC): an order which is sent with the price that the client wishes to trade at. It is executed immediately or rejected. It can be partially filled taking into account also MinQty <110>.

Market (MKT): an order which must be executed immediately at Wincent's best price, regardless of what it is.

New Order Single <D>

Message is used for FOK, IOC or MKT orders. The effective time tag can be specified to set an expiry time to protect against severe latency or accidental resubmission of an order.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>New Order Single <D></u>
<u>ClOrdId <11></u>	Y	Unique identifier assigned by the client to this order (e.g. UUIDv4)
<u>MinQty <110></u>	N	If <u>TimeInForce <59></u> is 3 (= IOC), sets the minimum quantity of an order to be executed
<u>Symbol <55></u>	Y	Symbol of the market data entry
<u>Side <54></u>	Y	The side of the order
<u>TransactTime <60></u>	Y	Time this order request was initiated by the trader, trading system or intermediary
<u>OrderQty <38></u>	Y	The quantity in base asset
<u>OrdType <40></u>	Y	Order type. Valid values: 1 (= Market) or 2 (= Limit)
<u>Price <44></u>	C	If <u>OrdType <40></u> is 2 in quote asset for a unit of the base asset

Tag	Required	Description
<u>QuoteID <117></u>	N	Should be set if the order is a response to a specific quote from <u>Market Data Snapshot/Full Refresh <W></u> matching <u>QuoteEntryID <299></u>
<u>TimeInForce <59></u>	C	If <u>OrdType <40></u> is 2 (= Limit). Valid values: 3 (= IOC) or 4 (= FOK)
<u>EffectiveTime <168></u>	N	The time at which to expire this order in UTC. Same format as expected as <u>SendingTime <52></u> . Used only to protect against sever latency or unforeseen accidents.
Standard Trailer	Y	

Execution Report <8>

The message is issued in response to a valid New Order Single <D> message. First, the API will confirm the receipt of an order in an execution report and after processing the order it will relay a rejected or filled status through another execution report.

Tag	Required	Description
Standard Header	Y	<u>MsgType <35></u> = <u>Execution Report <8></u>
<u>OrderID <37></u>	Y	Unique identifier assigned by Wincent to the order
<u>ClOrdId <11></u>	Y	Unique identifier assigned by the client to this order echoed back from <u>New Order Single <D></u>
<u>ExecId <17></u>	Y	Unique identifier of the execution messages
<u>ExecType <150></u>	Y	Describes the purpose of the execution report. Valid values: 0 (= New), 8 (= Rejected), C (= Expired) and F (= Trade). C (= Expired) is used only if <u>EffectiveTime <168></u> is set.
<u>OrdStatus <39></u>	Y	Describes the current order status. Valid values: 0 (= New), 1 (= Partially filled), 2 (= Filled), 8 (= Rejected), C (= Expired)
<u>Symbol <55></u>	Y	Symbol of the market data entry
<u>Side <54></u>	Y	The side of the order
<u>OrderQty <38></u>	Y	Order quantity echoed back
<u>OrdType <40></u>	Y	Order type echoed back
<u>Price <44></u>	C	If <u>OrdType <40></u> is 2 (= Limit), echoed back from the order
<u>TimeInForce <59></u>	C	If <u>OrdType <40></u> is 2 (= Limit), echoed back from the order
<u>EffectiveTime <168></u>	N	The time at which to expire this order in UTC, echoed back from the order
<u>LastQty <32></u>	C	If <u>ExecType <150></u> is F (= Trade) contains the filled quantity

Tag	Required	Description
<u>LastPx <31></u>	C	If <u>ExecType <150></u> is F (= Trade) contains the executed price
<u>LeavesQty <151></u>	Y	Quantity that is cancelled from the order if <u>OrdStatus <39></u> is 1 (= Partially filled), otherwise 0 <u>LeavesQty <151></u> = <u>OrderQty <38></u> - <u>CumQty <14></u>
<u>CumQty <14></u>	Y	Currently executed quantity
<u>AvgPx <6></u>	Y	The average price of fills on this order. 0 for rejected orders
<u>MinQty <110></u>	N	If <u>TimeInForce <59></u> is 3 (= IOC), echoed back from the order
<u>TransactTime <60></u>	Y	Time the transaction represented by this ExecutionReport occurred (in UTC)
<u>Text <58></u>	N	May be present if <u>ExecType <150></u> is 8 (= Rejected)
Standard Trailer	Y	

This page was built using the Antora default UI.

The source code for this UI is licensed under the terms of the MPL-2.0 license.