Kraken Websockets API

1.9.2

Kraken Websockets API 1.9.2

Overview

WebSockets API offers real-time market data updates. WebSockets is a bidirectional protocol offering fastest real-time data, helping you build real-time applications. The public message types presented below do not require authentication. Private-data messages can be subscribed on a separate authenticated endpoint.

Your use of the Kraken WebSockets API is subject to the Kraken Terms & Conditions, Privacy Notice, as well as all other applicable terms and disclosures made available on www.kraken.com.

General Considerations

- TLS with SNI (Server Name Indication) is required in order to establish a Kraken WebSockets API connection. See Cloudflare's "What is SNI?" guide for more details.
- All messages sent and received via WebSockets are encoded in JSON format
- All decimal fields (including timestamps) are quoted to preserve precision.
- Timestamps should not be considered unique and not be considered as aliases for transaction IDs. Also, the granularity of timestamps is not representative of transaction rates.
- At least one private message should be subscribed to keep the authenticated client connection open.
- Please use REST API endpoint AssetPairs to fetch the list of pairs which can be subscribed via WebSockets API. For example, field 'wsname' gives the supported pairs name which can be used to subscribe.
- Cloudflare imposes a connection/re-connection rate limit (per IP address) of approximately 150 attempts per rolling 10 minutes. If this is exceeded, the **IP is banned for 10 minutes.**
- **Recommended reconnection behaviour** is to **(1)** attempt reconnection instantly up to a handful of times if the websocket is dropped randomly during normal operation but **(2)** after maintenance or extended downtime, attempt to reconnect no more quickly than once every 5 seconds. There is no advantage to reconnecting more rapidly after maintenance during `cancel_only` mode.

Connection details

Connection details for production environment:

URL	Description
wss://ws.kraken.com/	Once the socket is open you can subscribe to a public channel by sending a subscribe request message.

Connection details for authenticated production access:

URL	Description
wss://ws-auth.kraken.com/	Once the socket is open you can subscribe to private-data channels by sending an authenticated subscribe request message.

Connection details for Beta environment:

URL	Description
wss://beta-ws.kraken.com/	Once the socket is open you can subscribe to a public channel by sending a subscribe request message. Websockets Beta Documentation

Connection details for authenticated Beta access:

URL	Description
wss://beta-ws-auth.kraken.com/	Once the socket is open you can subscribe to private-data channels by sending an authenticated subscribe request message. Websockets Beta Documentation

Authentication

The API client must request an authentication "token" via the following REST API endpoint "GetWebSocketsToken" to connect to WebSockets Private endpoints. The token should be used within 15 minutes of creation. The token does not expire once a connection to a WebSockets API private message (openOrders or ownTrades) is maintained.

Endpoint URL: https://api.kraken.com/0/private/GetWebSocketsToken

The resulting token must be provided in the "token" field of any new private WebSocket feed subscription:

```
{
  "event": "subscribe",
  "subscription":
  {
    "name": "ownTrades",
    "token": "WW91ciBhdXRoZW50aWNhdGlvbiB0b2tlbiBnb2VzIGhlcmUu"
  }
}
```

Book Checksum

Each book update message will have a checksum value appended. The checksum is a CRC32 value based on the top 10 bids and 10 asks, and you can use it to verify that your data is correct and up to date by calculating the checksum independently and comparing it against the value provided.

Checksums will not be sent in book snapshot messages, but rather only in book update messages. In the following sample book update messages, note the checksum field appears in the last bid or ask map structure in the message.

Sample with asks only:

```
[
    0,
    {
        "a": [
            ["0.05120", "0.00000500", "1582905486.493008"],
            ["0.05275", "0.00000500", "1582905486.493034"]
        ],
        "c": "974947235" <-- CRC32 checksum is here.
    },
    "book-1000",
    "XBT/USD"
]
```

Sample with bids only:

```
[
    0,
    {
        "b": [
            ["0.04765", "0.00000500", "1582905486.493008"],
            ["0.04940", "0.00000500", "1582905486.493034"]
        ],
        "c": "974947235" <-- CRC32 checksum is here.
    },
    "book-1000",
    "XBT/USD"
]
```

Sample with both bids and asks:

```
[
    0,
    {
        "a": [
            ["0.05120", "0.00000500", "1582905486.493008"],
            ["0.05275", "0.00000500", "1582905486.493034"]
        ]
    },
    {
        "b": [
            ["0.04765", "0.00000500", "1582905486.493008"],
            ["0.04940", "0.00000500", "1582905486.493034"]
        ],
        "c": "974947235" <-- CRC32 checksum is here.
    },
    "book-1000",
    "XBT/USD"
]
```

Book Checksum Calculation

The checksum is computed by concatenating the top 10 bids and asks in the current book in a particular format and then taking the CRC32 checksum of that string. The price and volume values should be treated as a string and formatted and concatenated as follows.

Note: Processing order is important. First, the top ten ask price levels should be processed, sorted by price from low to high. Then, the top ten bid price levels should be processed, sorted by price from high to low. Note, the price levels will be received in the correct sort order from the exchange.

Consider you are subscribed at depth = 100 on the "book" channel and you receive the following book update message:

```
    [ "0.05000", "0.00000304", "1582905487.439814" ]
```

This update should be processed as follows:

1. Apply the update to your local copy of the book. The price level updates should be processed in the sequence of the array provided by the exchange. The sequence of the array is important - the last entries in array are the most recent. Do not sort by timestamp, as there can be multiple price level updates in a microsecond period.
2. For each of the top ten ask price levels, sorted by price from low to high:
    1. Remove the decimal character, '.', from the price, i.e. "0.05000" -> "005000".
    2. Remove all leading zero characters from the price. i.e. "005000" -> "5000".
    3. Add the formatted price string to the concatenation.
    4. Repeat steps a-c above but for the volume.

    For example, the price level corresponding to the sample update above would be formatted as "5000304". Note the timestamp values are not used in this calculation.

3. Repeat the above steps for the top ten bids, sorted by price from high to low.
4. Feed the concatenated string as input to a CRC32 checksum function, storing the result.

5. Cast the result (comprising 32 bits) as an unsigned 32-bit integer. This value can now be compared to the checksum received to ensure your local book is accurate.

For example, given the following book state:

```
{
"as": [
    [ "0.05005", "0.00000500", "1582905487.684110" ],
    [ "0.05010", "0.00000500", "1582905486.187983" ],
    [ "0.05015", "0.00000500", "1582905484.480241" ],
    [ "0.05020", "0.00000500", "1582905486.645658" ],
    [ "0.05025", "0.00000500", "1582905486.859009" ],
    [ "0.05030", "0.00000500", "1582905488.601486" ],
    [ "0.05035", "0.00000500", "1582905488.357312" ],
    [ "0.05040", "0.00000500", "1582905488.785484" ],
    [ "0.05045", "0.00000500", "1582905485.302661" ],
    [ "0.05050", "0.00000500", "1582905486.157467" ] ],
"bs": [
    [ "0.05000", "0.00000500", "1582905487.439814" ],
    [ "0.04995", "0.00000500", "1582905485.119396" ],
    [ "0.04990", "0.00000500", "1582905486.432052" ],
    [ "0.04980", "0.00000500", "1582905480.609351" ],
    [ "0.04975", "0.00000500", "1582905476.793880" ],
    [ "0.04970", "0.00000500", "1582905486.767461" ],
    [ "0.04965", "0.00000500", "1582905481.767528" ],
    [ "0.04960", "0.00000500", "1582905487.378907" ],
    [ "0.04955", "0.00000500", "1582905483.626664" ],
    [ "0.04950", "0.00000500", "1582905488.509872" ] ]
}
```

The checksum input should be as follows (newlines appear here for convenience only and should not be included):

```
"5005500501050050155005020500502550O
50305005035500504050050455005050500
50005004995500499050049805004975500
49705004965500496050049555004950500"
```

The final unsigned CRC32 checksum value will then be "974947235".

Sequence Numbers

The private feeds "openOrders" and "ownTrades" both contain sequence numbers ("sequence") in their messages. These numbers are monotonically increasing integers, beginning at 1, that operate on a per-connection and per-feed basis. These are meant to help clients identify if they are, for any reason, dropping messages or receiving/processing messages in a different order than they were sent from our servers. Example payloads can be seen in the openOrders and ownTrades sections below.

Error Types

The following error messages are thrown as part of subscriptionStatus message.

> The following error messages may be thrown for public data requests.
>
> - Already subscribed
> - Currency pair not in ISO 4217-A3 format
> - Malformed request
> - Pair field must be an array
> - Pair field unsupported for this subscription type
> - Pair(s) not found
> - Subscription book depth must be an integer
> - Subscription depth not supported
> - Subscription field must be an object
> - Subscription name invalid
> - Subscription object unsupported field
> - Subscription ohlc interval must be an integer
> - Subscription ohlc interval not supported
> - Subscription ohlc requires interval
>
> The following error messages may be thrown for private data requests. Segments in brackets ([]) indicate additional information that may or not be present in the error message.
>
> - EAccount:Invalid permissions
> - EAuth:Account temporary disabled
> - EAuth:Account unconfirmed
> - EAuth:Rate limit exceeded
> - EAuth:Too many requests
> - EDatabase: Internal error (to be deprecated)
> - EGeneral:Internal error[:<code>]
> - EGeneral:Invalid arguments
> - EOrder:Cannot open opposing position
> - EOrder:Cannot open position
> - EOrder:Insufficient funds (insufficient user funds)
> - EOrder:Insufficient margin (exchange does not have sufficient funds to allow margin trading)
> - EOrder:Invalid price
> - EOrder:Margin allowance exceeded
> - EOrder:Margin level too low
> - EOrder:Margin position size exceeded (client would exceed the maximum position size for this pair)
> - EOrder:Order minimum not met (volume too low)
> - EOrder:Orders limit exceeded
> - EOrder:Positions limit exceeded
> - EOrder:Rate limit exceeded
> - EOrder:Scheduled orders limit exceeded

- EOrder:Unknown position
- EService:Deadline elapsed
- EService:Market in cancel_only mode
- EService:Market in limit_only mode
- EService:Market in post_only mode
- EService:Unavailable
- ETrade:Invalid request

General error messages will have the following structure. However, error responses related to particular requests such as subscriptionStatus, addOrder, cancelOrder may be returned in the appropriate response message type:

- subscriptionStatus for subscribe and unsubscribe requests,
- addOrderStatus for addOrder requests, and
- cancelOrderStatus for cancelOrder requests

Name	**Type**	**Description**
event	string	error
errorMessage	string	Error detail message.
reqid	integer	Optional - client originated ID reflected in response message

Examples of payload

```
{
  "errorMessage": "Malformed request",
  "event": "error"
}
```

```
{
  "errorMessage":"Exceeded msg rate",
  "event": "error",
  "reqid": 42
}
```

Example API Clients

Below is sample code that can be referenced when writing your own API client. Please keep in mind that Payward nor the third party authors are responsible for losses due to bugs or improper use of the APIs. Payward has performed an initial review of the safety of the third party code before listing them but cannot vouch for any changes added since then. If you have concerns, please contact support.

- **Go**
  jurijbajzelj/kraken_ws_orderbook
- **Python**
  krakenfx/kraken-wsclient-py

Changelog

2024-01-10

- Websockets 1.9.2 released
- Range of valid offsets (from currrent time) for the "deadline" parameter changed to 500 milliseconds to 60 seconds, default is 5 seconds.

2023-03-02

- Websockets 1.9.1 released
- Added contingent field on private open order feed
- On public instances during trading engine maintenance we are going to keep clients connected and disconnect them once when we can process trading engine updates.

2022-03-22

- Websockets 1.9.0 released
- Support for editOrder added

2021-03-31

- Websockets 1.8.3 released
- Support for addOrder "deadline" added

2021-02-25

- Websockets 1.8.0 released
- Support for "timeinforce" and Immediate-or-Cancel (IOC) added

2021-02-04

- Websockets 1.7.2 released
- Improve public market data snapshot performance
- Change close code to 1008 (Policy Violation) from 1013 for maximum number of connections, message rate limit, and slow websocket consumption

- Add a policy rule for the maximum rate of subscriptions
- Add a new generic error type with (optional) internal error codes, EGeneral:Internal Error[:<code>]

2021-01-30

Add:

- Websockets 1.7.0 released
- Userref field added in openOrders, ownTrades update messages

2021-01-13

Add:

- Dead man’s switch (cancelAllOrdersAfter) REST endpoint added

Fix:

- Intermittent public data websocket feed latency and connection instability issue resolved

2020-12-21

Add:

- Websockets 1.6.0 released
- Dead man’s switch (cancelAllOrdersAfter) functionality
- Post_only trading mode introduced for maintenance procedure (systemStatus)

2020-12-05

Add:

- Websockets 1.5.0 released
- Optional boolean \`ratecounter\` argument for openOrders subscription
- \`maxratecount\` and current \`ratecount\` reporting on openOrders feed
- Cancel_only trading mode introduced and reflected via \`systemStatus\` updates
- SystemStatus REST endpoint added

Change:

- Relaxed slow-consumer constraint on WS affecting some java client libraries

Fix:

- Maintain private WS connections during maintenance
- Public market data snapshot/stream synchronisation improvements
- Inactive/unimplemented order types removed from REST docs

2020-11-18

Add:

- CancelAll REST endpoint added

2020-11-02

Add:

- Websockets 1.4.0 released
- 'cancelAll' trading request functionality

Change:

- Performance upgrade to cancelOrder request handling
- Improve messages and close codes when killing WS connections

Fix:

- Return correct error for addOrder with invalid pair

2020-10-27

Add:

- Websockets 1.3.0 released
- Sequence numbers added on private (openOrders, ownTrades) feeds

2020-10-12

Change:

- Eliminated trading rate limit penalty for filled orders
- Performance improvement for REST real-time and historical market data endpoints

Fix:

- Reject 'market' conditional close orders
- Intermittent bug affecting Ticker REST endpoint resolved

2020-08-31

Add:

- Websockets 1.2.0 released
- cancel_reason added to openOrders stream
- Optional boolean parameter "snapshot" added for ownTrades feed

2020-08-04

Change:

- Minimum order sizes updated for 8 assets / 30 pairs

2020-07-16

Add:

- Add ‘maintenance’ as possible systemStatus message

Fix:

- Connection stability improvements

2020-06-22

Add:

- Public Websockets 1.1.0 released
- Order book ‘checksum’ added

2020-03-18

Fix:

- Reject ‘viqc’ order flag
- Handle ‘validate’ field appropriately
- Include ‘reqid’ with all error responses
- Stability improvements

2020-02-18

Add:

- Private websockets 1.0.0 released to production
- addOrder, cancelOrder trading requests introduced

2019-10-01

Add:

- Private websockets 0.3.0 in beta
- openOrders, ownTrades streams introduced

2019-02-04

Change:

- Public WS 1.00.01 released to production

Fix:

- Fix publishing of deleted price levels

2019-01-23

Add:

- Websockets public market data sandbox 0.1.1 released
- connectionID field added to systemStatus message

2019-01-18

Add:

- Websockets public market data sandbox 0.0.6 released
- ‘open’ prices on ohlc include 24-hour values

Change:

- Timestamp precision increased to microseconds for ohlc, spread, book, trade
- Sandbox URL change

2018-12-24

Change:

- Websockets public market data sandbox 0.0.5 released
- Timestamp field changed to string type

2018-12-07

Add:

- Websockets public market data sandbox 0.0.4 released

Change:

- Timestamp precision changed to milliseconds for ohlc, trade, spread, book feeds

2018-11-28

Add:

- Websockets public market data sandbox 0.0.3 released

Messages

ping

Request. Client can ping server to determine whether connection is alive, server responds with pong. This is an application level ping as opposed to default ping in websockets standard which is server initiated

Payload

Name	Type	Description
event	string	
reqid	integer	Optional - client originated ID reflected in response message

Example of payload

```
{
  "event": "ping",
  "reqid": 42
}
```

pong

Response. Server pong response to a ping to determine whether connection is alive. This is an application level pong as opposed to default pong in websockets standard which is sent by client in response to a ping

Payload

Name	Type	Description
event	string	
reqid	integer	Optional - matching client originated request ID

Example of payload

```
{
  "event": "pong",
  "reqid": 42
}
```

heartbeat

Publication: Server heartbeat sent if no subscription traffic within 1 second (approximately)

Payload

Name	Type	Description
event	string	

Example of payload

```
{
  "event": "heartbeat"
}
```

systemStatus

Publication: Status sent on connection or system status changes.

Payload

Name	Type	Description
connectionID	integer	Optional - Connection ID (will appear only in initial connection status message)
event	string	systemStatus
status	string	online|maintenance|cancel_only|limit_only|post_only
version	string	

Example of payload

```
{
  "connectionID": 8628615390848610000,
  "event": "systemStatus",
  "status": "online",
  "version": "1.0.0"
}
```

subscribe

Request. Subscribe to a topic on a single or multiple currency pairs.

Payload

Name	Type	Description
event	string	subscribe
reqid	integer	Optional - client originated ID reflected in response message
pair	array	Optional - Array of currency pairs. Format of each pair is "A/B", where A and B are ISO 4217-

Name	Type	Description
		A3 for standardized assets and popular unique symbol if not standardized.
subscription	object	
depth	integer	Optional - depth associated with book subscription in number of levels each side, default 10. Valid Options are: 10, 25, 100, 500, 1000
interval	integer	Optional - Time interval associated with ohlc subscription in minutes. Default 1. Valid Interval values: 1|5|15|30|60|240|1440|10080|21600
name	string	book|ohlc|openOrders|ownTrades|spread|ticker|trade|*, * for all available channels depending on the connected environment
ratecounter	boolean	Optional - whether to send rate-limit counter in updates (supported only for openOrders subscriptions; default = false)
snapshot	boolean	Optional - whether to send historical feed data snapshot upon subscription (supported only for ownTrades subscriptions; default = true)
token	string	Optional - base64-encoded authentication token for private-data endpoints
consolidate_taker	boolean	Optional - for ownTrades, whether to consolidate order fills by root taker trade(s), default = true. If false, all order fills will show separately.

Example of payload

```
{
  "event": "subscribe",
  "pair": [
    "XBT/USD",
    "XBT/EUR"
  ],
  "subscription": {
    "name": "ticker"
  }
}
```

```
{
  "event": "subscribe",
  "pair": [
    "XBT/EUR"
  ],
  "subscription": {
    "interval": 5,
    "name": "ohlc"
  }
}
```

```
{
  "event": "subscribe",
  "subscription": {
    "name": "ownTrades",
    "token": "WW91ciBhdXRoZW50aWNhdGlvbiB0b2tlbiBnb2VzIGhlcmUu"
  }
}
```

unsubscribe

Request. Unsubscribe, can specify a channelID or multiple currency pairs.

Payload

Name	**Type**	**Description**
event	string	unsubscribe
reqid	integer	Optional - client originated ID reflected in response message
pair	array	Optional - Array of currency pairs. Format of each pair is "A/B", where A and B are ISO 4217-A3 for standardized assets and popular unique symbol if not standardized.
subscription	object	
depth	integer	Optional - depth associated with book subscription in number of levels each side, default 10. Valid Options are: 10, 25, 100, 500, 1000
interval	integer	Optional - Time interval associated with ohlc subscription in minutes. Default 1. Valid Interval values: 1|5|15|30|60|240|1440|10080|21600
name	string	book|ohlc|openOrders|ownTrades|spread|ticker|trade|*, * for all available channels depending on the connected environment
token	string	Optional - base64-encoded authentication token for private-data endpoints

Example of payload

```
{
  "event": "unsubscribe",
  "pair": [
    "XBT/EUR",
    "XBT/USD"
  ],
  "subscription": {
    "name": "ticker"
  }
}
```

```
{
  "channelID": 10001,
  "event": "unsubscribe"
}
```

```
{
  "event": "unsubscribe",
  "subscription": {
    "name": "ownTrades",
    "token": "WW91ciBhdXRoZW50aWNhdGlvbiB0b2tlbiBnb2VzIGhlcmUu"
  }
}
```

subscriptionStatus

Response. Subscription status response to subscribe, unsubscribe or exchange initiated unsubscribe.

Payload

Name	Type	Description
channelName	string	Channel Name on successful subscription. For payloads 'ohlc' and 'book', respective interval or depth will be added as suffix.
event	string	
reqid	integer	Optional - matching client originated request ID
pair	string	Optional - Currency pair, applicable to public messages only
status	string	Status of subscription
subscription	object	
depth	integer	Optional - depth associated with book subscription in number of levels each side, default 10. Valid Options are: 10, 25, 100, 500, 1000
interval	integer	Optional - Time interval associated with ohlc subscription in minutes. Default 1. Valid Interval values: 1|5|15|30|60|240|1440|10080|21600
maxratecount	integer	Optional - max rate-limit budget. Compare to the ratecounter field in the openOrders updates to check whether you are approaching the rate limit.
name	string	book|ohlc|openOrders|ownTrades|spread|ticker|trade|*, * for all available channels depending on the connected environment
token	string	Optional - base64-encoded authentication token for private-data endpoints
OneOf	oneOf	
errorMessage	string	Error message
channelID	integer	

Name	Type	Description
		Channel ID on successful subscription, applicable to public messages only - deprecated, use channelName and pair

Example of payload

```
{
  "channelID": 10001,
  "channelName": "ticker",
  "event": "subscriptionStatus",
  "pair": "XBT/EUR",
  "status": "subscribed",
  "subscription": {
    "name": "ticker"
  }
}

{
  "channelID": 10001,
  "channelName": "ohlc-5",
  "event": "subscriptionStatus",
  "pair": "XBT/EUR",
  "reqid": 42,
  "status": "unsubscribed",
  "subscription": {
    "interval": 5,
    "name": "ohlc"
  }
}

{
  "channelName": "ownTrades",
  "event": "subscriptionStatus",
  "status": "subscribed",
  "subscription": {
    "name": "ownTrades"
  }
}

{
  "errorMessage": "Subscription depth not supported",
  "event": "subscriptionStatus",
  "pair": "XBT/USD",
  "status": "error",
  "subscription": {
    "depth": 42,
    "name": "book"
  }
}
```

ticker

Publication: Ticker information on currency pair.

Payload

Name	Type	Description
channelID	integer	Channel ID of subscription - deprecated, use channelName and pair
(Anonymous)	object	
a	array	Ask
price	decimal	Best ask price
wholeLotVolume	integer	Whole lot volume
lotVolume	decimal	Lot volume
b	array	Bid
price	decimal	Best bid price
wholeLotVolume	integer	Whole lot volume
lotVolume	decimal	Lot volume
c	array	Close
price	decimal	Price
lotVolume	decimal	Lot volume
v	array	Volume
today	decimal	Value today
last24Hours	decimal	Value over last 24 hours
p	array	Volume weighted average price
today	decimal	Value today
last24Hours	decimal	Value over last 24 hours
t	array	Number of trades
today	integer	Value today
last24Hours	integer	Value over last 24 hours
l	array	Low price
today	decimal	Value today
last24Hours	decimal	Value over last 24 hours
h	array	High price
today	decimal	Value today
last24Hours	decimal	Value over last 24 hours
o	array	Open Price
today	decimal	Value today
last24Hours	decimal	Value over last 24 hours
channelName	string	Channel Name of subscription
pair	string	Asset pair

Example of payload

```
[
  0,
  {
    "a": [
      "5525.40000",
      1,
      "1.000"
    ],
    "b": [
      "5525.10000",
      1,
      "1.000"
    ],
    "c": [
      "5525.10000",
      "0.00398963"
    ],
    "h": [
      "5783.00000",
      "5783.00000"
    ],
    "l": [
      "5505.00000",
      "5505.00000"
    ],
    "o": [
      "5760.70000",
      "5763.40000"
    ],
    "p": [
      "5631.44067",
      "5653.78939"
    ],
    "t": [
      11493,
      16267
    ],
    "v": [
      "2634.11501494",
      "3591.17907851"
    ]
  },
  "ticker",
  "XBT/USD"
]
```

ohlc

Publication: Open High Low Close (Candle) feed for a currency pair and interval period.

Description: When subscribed for OHLC, a snapshot of the last valid candle (irrespective of the endtime) will be sent, followed by updates to the running candle. For example, if a subscription is made to 1 min candle and there have been no trades for 5 mins, a snapshot of the last 1 min candle from 5 mins ago will be published. The endtime can be used to determine that it is an old candle.

Payload

Name	Type	Description
channelID	integer	Channel ID of subscription - deprecated, use channelName and pair
Array	array	
time	decimal	Candle last update time, in seconds since epoch
etime	decimal	End time of interval, in seconds since epoch
open	decimal	Open price of interval
high	decimal	High price within interval
low	decimal	Low price within interval
close	decimal	Close price of interval
vwap	decimal	Volume weighted average price within interval
volume	decimal	Accumulated volume within interval
count	integer	Number of trades within interval
channelName	string	Channel Name of subscription
pair	string	Asset pair

Example of payload

```
[
  42,
  [
    "1542057314.748456",
    "1542057360.435743",
    "3586.70000",
    "3586.70000",
    "3586.60000",
    "3586.60000",
    "3586.68894",
    "0.03373000",
    2
  ],
  "ohlc-5",
  "XBT/USD"
]
```

trade

Publication: Trade feed for a currency pair.

Payload

Name	Type	Description
channelID	integer	Channel ID of subscription - deprecated, use channelName and pair
Array	array	
Array	array	
price	decimal	Price
volume	decimal	Volume
time	decimal	Time, seconds since epoch
side	string	Triggering order side, buy/sell
orderType	string	Triggering order type market/limit
misc	string	Miscellaneous
channelName	string	Channel Name of subscription
pair	string	Asset pair

Example of payload

```
[
  0,
  [
    [
      "5541.20000",
      "0.15850568",
      "1534614057.321597",
      "s",
      "l",
      ""
    ],
    [
      "6060.00000",
      "0.02455000",
      "1534614057.324998",
      "b",
      "l",
      ""
    ]
  ],
  "trade",
  "XBT/USD"
]
```

spread

Publication: Spread feed for a currency pair.

Payload

Name	Type	Description
channelID	integer	Channel ID of subscription - deprecated, use channelName and pair
Array	array	
bid	decimal	Bid price
ask	decimal	Ask price
timestamp	decimal	Time, seconds since epoch
bidVolume	decimal	Bid Volume
askVolume	decimal	Ask Volume
channelName	string	Channel Name of subscription
pair	string	Asset pair

Example of payload

```
[
  0,
  [
    "5698.40000",
    "5700.00000",
    "1542057299.545897",
    "1.01234567",
    "0.98765432"
  ],
  "spread",
  "XBT/USD"
]
```

book

Publication: Order book levels. On subscription, a snapshot will be published at the specified depth, following the snapshot, level updates will be published

Snapshot payload

Name	Type	Description
channelID	integer	Channel ID of subscription - deprecated, use channelName and pair
(Anonymous)	object	
as	array	Array of price levels, ascending from best ask

Name	**Type**	**Description**
Array	array	Anonymous array of level values
price	decimal	Price level
volume	decimal	Price level volume, for updates volume = 0 for level removal/deletion
timestamp	decimal	Price level last updated, seconds since epoch
bs	array	Array of price levels, descending from best bid
Array	array	Anonymous array of level values
price	decimal	Price level
volume	decimal	Price level volume, for updates volume = 0 for level removal/deletion
timestamp	decimal	Price level last updated, seconds since epoch
channelName	string	Channel Name of subscription
pair	string	Asset pair

Example of snapshot payload

```
[
  0,
  {
    "as": [
      [
        "5541.30000",
        "2.50700000",
        "1534614248.123678"
      ],
      [
        "5541.80000",
        "0.33000000",
        "1534614098.345543"
      ],
      [
        "5542.70000",
        "0.64700000",
        "1534614244.654432"
      ]
    ],
    "bs": [
      [
        "5541.20000",
        "1.52900000",
        "1534614248.765567"
      ],
      [
        "5539.90000",
        "0.30000000",
        "1534614241.769870"
      ],
      [
        "5539.50000",
        "5.00000000",
        "1534613831.243486"
      ]
    ]
  },
  "book-100",
  "XBT/USD"
]
```

Update payload

Name	Type	Description
channelID	integer	Channel ID of subscription - deprecated, use channelName and pair
AnyOf	anyOf	
(Anonymous)	object	Container for ask updates
a	array	Ask array of level updates
(Array)	array	Anonymous array of level values
price	decimal	Price level
volume	decimal	Price level volume, for updates volume = 0 for level removal/deletion
timestamp	decimal	Price level last updated, seconds since epoch
updateType	string	Optional - "r" in case update is a republished update
c	string	Optional - Book checksum as a quoted unsigned 32-bit integer, present only within the last update container in the message. See calculation details.
(Anonymous)	object	Container for bid updates
b	array	Bid array of level updates
(Array)	array	Anonymous array of level values
price	decimal	Price level
volume	decimal	Price level volume, for updates volume = 0 for level removal/deletion
timestamp	decimal	Price level last updated, seconds since epoch
updateType	string	Optional - "r" in case update is a republished update
c	string	Optional - Book checksum as a quoted unsigned 32-bit integer, present only within the last update container in the message. See calculation details.
channelName	string	Channel Name of subscription
pair	string	Asset pair

Example of update payload

```
[
  1234,
  {
    "a": [
      [
        "5541.30000",
        "2.50700000",
        "1534614248.456738"
      ],
      [
        "5542.50000",
        "0.40100000",
        "1534614248.456738"
      ]
    ],
    "c": "974942666"
  },
  "book-10",
  "XBT/USD"
]

[
  1234,
  {
    "b": [
      [
        "5541.30000",
        "0.00000000",
        "1534614335.345903"
      ]
    ],
    "c": "974942666"
  },
  "book-10",
  "XBT/USD"
]

[
  1234,
  {
    "a": [
      [
        "5541.30000",
        "2.50700000",
        "1534614248.456738"
      ],
      [
        "5542.50000",
        "0.40100000",
        "1534614248.456738"
      ]
    ]
  },
  {
    "b": [
      [
        "5541.30000",
        "0.00000000",
        "1534614335.345903"
      ]
    ],
    "c": "974942666"
  },
  "book-10",
  "XBT/USD"
]
```

Example of republish payload

```
[
  1234,
  {
    "a": [
      [
        "5541.30000",
        "2.50700000",
        "1534614248.456738",
        "r"
      ],
      [
        "5542.50000",
        "0.40100000",
        "1534614248.456738",
        "r"
      ]
    ],
    "c": "974942666"
  },
  "book-25",
  "XBT/USD"
]
```

ownTrades

Publication: Own trades. On subscription last 50 trades for the user will be sent, followed by new trades.

Payload

Name	Type	Description
(Dictionary)	object	
tradeid	object	Trade object
ordertxid	string	order responsible for execution of trade
postxid	string	Position trade id

Name	Type	Description
pair	string	Asset pair
time	decimal	unix timestamp of trade
type	string	type of order (buy/sell)
ordertype	string	order type
price	decimal	average price order was executed at (quote currency)
cost	decimal	total cost of order (quote currency)
fee	decimal	total fee (quote currency)
vol	decimal	volume (base currency)
margin	decimal	initial margin (quote currency)
userref	integer	user reference ID
channelName	string	Channel Name of subscription
(Anonymous)	object	
sequence	integer	sequence number for ownTrades subcription

Example of payload

```
[
  [
    {
      "TDLH43-DVQXD-2KHVYY": {
        "cost": "1000000.00000",
        "fee": "1600.00000",
        "margin": "0.00000",
        "ordertxid": "TDLH43-DVQXD-2KHVYY",
        "ordertype": "limit",
        "pair": "XBT/EUR",
        "postxid": "OGTT3Y-C6I3P-XRI6HX",
        "price": "100000.00000",
        "time": "1560516023.070651",
        "type": "sell",
        "vol": "1000000000.00000000"
      }
    },
    {
      "TDLH43-DVQXD-2KHVYY": {
        "cost": "1000000.00000",
        "fee": "600.00000",
        "margin": "0.00000",
        "ordertxid": "TDLH43-DVQXD-2KHVYY",
        "ordertype": "limit",
        "pair": "XBT/EUR",
        "postxid": "OGTT3Y-C6I3P-XRI6HX",
        "price": "100000.00000",
        "time": "1560516023.070658",
        "type": "buy",
        "vol": "1000000000.00000000"
      }
    },
    {
      "TDLH43-DVQXD-2KHVYY": {
        "cost": "1000000.00000",
        "fee": "1600.00000",
        "margin": "0.00000",
        "ordertxid": "TDLH43-DVQXD-2KHVYY",
        "ordertype": "limit",
        "pair": "XBT/EUR",
        "postxid": "OGTT3Y-C6I3P-XRI6HX",
        "price": "100000.00000",
        "time": "1560520332.914657",
        "type": "sell",
        "vol": "1000000000.00000000"
      }
    },
    {
      "TDLH43-DVQXD-2KHVYY": {
        "cost": "1000000.00000",
        "fee": "600.00000",
        "margin": "0.00000",
        "ordertxid": "TDLH43-DVQXD-2KHVYY",
        "ordertype": "limit",
        "pair": "XBT/EUR",
        "postxid": "OGTT3Y-C6I3P-XRI6HX",
        "price": "100000.00000",
        "time": "1560520332.914664",
        "type": "buy",
        "vol": "1000000000.00000000"
      }
    }
  ],
  "ownTrades",
  {
    "sequence": 2948
  }
]
```

openOrders

Publication: Open orders. Feed to show all the open orders belonging to the authenticated user. Initial snapshot will provide list of all open orders and then any updates to the open orders list will be sent. For status change updates, such as 'closed', the fields `orderid` and `status` will be present in the payload.

The following order cancel reasons may appear in the openOrders feed:

- Cannot trade with self
- Order replaced

- Post only order
- User requested

Payload

Name	Type	Description
(Dictionary)	object	
orderid	object	Order object
refid	string	Referral order transaction id that created this order
userref	integer	user reference ID
status	string	status of order
opentm	decimal	unix timestamp of when order was placed
starttm	decimal	unix timestamp of order start time (if set)
display_volume	decimal	Optional dependent on whether order type is iceberg - the visible quantity for iceberg order types
display_volume_remain	decimal	Optional dependent on whether order type is iceberg - the visible quantity remaing in the order for iceberg order types
expiretm	string	unix timestamp of order end time (if set)
contingent	object	conditional close order info (if conditional close set)
ordertype	string	conditional close order type
price	decimal	primary price of the conditional close order
price2	decimal	secondary price of the conditional close order
oflags	string	Optional - comma delimited list of order flags, of the conditional close order viqc = volume in quote currency (not currently available), fcib = prefer fee in base currency, fciq = prefer fee in quote currency, nompp = no market price protection, post = post only order (available when ordertype = limit)
descr	object	order description info
pair	string	asset pair
position	string	Optional - position ID (if applicable)
type	string	type of order (buy/sell)
ordertype	string	order type
price	decimal	primary price
price2	decimal	secondary price
leverage	decimal	amount of leverage
order	string	order description
close	string	conditional close order description (if conditional close set)

Name	Type	Description
lastupdated	decimal	unix timestamp of last change (for updates)
vol	decimal	volume of order (base currency unless viqc set in oflags)
vol_exec	decimal	total volume executed so far (base currency unless viqc set in oflags)
cost	decimal	total cost (quote currency unless unless viqc set in oflags)
fee	decimal	total fee (quote currency)
avg_price	decimal	average price (cumulative; quote currency unless viqc set in oflags)
stopprice	decimal	stop price (quote currency, for trailing stops)
limitprice	decimal	triggered limit price (quote currency, when limit based order type triggered)
misc	string	comma delimited list of miscellaneous info: stopped=triggered by stop price, touched=triggered by touch price, liquidation=liquidation, partial=partial fill
oflags	string	Optional - comma delimited list of order flags. viqc = volume in quote currency (not currently available), fcib = prefer fee in base currency, fciq = prefer fee in quote currency, nompp = no market price protection, post = post only order (available when ordertype = limit)
timeinforce	string	Optional - time in force.
cancel_reason	string	Optional - cancel reason, present for all cancellation updates (status="canceled") and for some close updates (status="closed")
ratecount	integer	Optional - rate-limit counter, present if requested in subscription request. See Trading Rate Limits.
channelName	string	Channel Name of subscription
(Anonymous)	object	
sequence	integer	sequence number for openOrders subcription

Example of payload

```
[
  [
    {
      "OGTT3Y-C6I3P-XRI6HX": {
        "avg_price": "34.50000",
        "cost": "0.00000",
        "descr": {
```

```
        "close": "",
        "leverage": "0:1",
        "order": "sell 10.00345345 XBT/EUR @ limit 34.50000 with 0:1 lev
        "ordertype": "limit",
        "pair": "XBT/EUR",
        "price": "34.50000",
        "price2": "0.00000",
        "type": "sell"
      },
      "expiretm": "0.000000",
      "fee": "0.00000",
      "limitprice": "34.50000",
      "misc": "",
      "oflags": "fcib",
      "opentm": "0.000000",
      "refid": "OKIVMP-5GVZN-Z2D2UA",
      "starttm": "0.000000",
      "status": "open",
      "stopprice": "0.000000",
      "userref": 0,
      "vol": "10.00345345",
      "vol_exec": "0.00000000"
    }
  },
  {
    "OGTT3Y-C6I3P-XRI6HX": {
      "avg_price": "5334.60000",
      "cost": "0.00000",
      "descr": {
        "close": "",
        "leverage": "0:1",
        "order": "sell 0.00000010 XBT/EUR @ limit 5334.60000 with 0:1 le
        "ordertype": "limit",
        "pair": "XBT/EUR",
        "price": "5334.60000",
        "price2": "0.00000",
        "type": "sell"
      },
      "expiretm": "0.000000",
      "fee": "0.00000",
      "limitprice": "5334.60000",
      "misc": "",
      "oflags": "fcib",
      "opentm": "0.000000",
      "refid": "OKIVMP-5GVZN-Z2D2UA",
      "starttm": "0.000000",
      "status": "open",
      "stopprice": "0.000000",
      "userref": 0,
      "vol": "0.00000010",
      "vol_exec": "0.00000000"
    }
```

```
        },
        {
          "OGTT3Y-C6I3P-XRI6HX": {
            "avg_price": "90.40000",
            "cost": "0.00000",
            "descr": {
              "close": "",
              "leverage": "0:1",
              "order": "sell 0.00001000 XBT/EUR @ limit 90.40000 with 0:1 leve
              "ordertype": "limit",
              "pair": "XBT/EUR",
              "price": "90.40000",
              "price2": "0.00000",
              "type": "sell"
            },
            "expiretm": "0.000000",
            "fee": "0.00000",
            "limitprice": "90.40000",
            "misc": "",
            "oflags": "fcib",
            "opentm": "0.000000",
            "refid": "OKIVMP-5GVZN-Z2D2UA",
            "starttm": "0.000000",
            "status": "open",
            "stopprice": "0.000000",
            "userref": 0,
            "vol": "0.00001000",
            "vol_exec": "0.00000000"
          }
        },
        {
          "OGTT3Y-C6I3P-XRI6HX": {
            "avg_price": "9.00000",
            "cost": "0.00000",
            "descr": {
              "close": "",
              "leverage": "0:1",
              "order": "sell 0.00001000 XBT/EUR @ limit 9.00000 with 0:1 lever
              "ordertype": "limit",
              "pair": "XBT/EUR",
              "price": "9.00000",
              "price2": "0.00000",
              "type": "sell"
            },
            "expiretm": "0.000000",
            "fee": "0.00000",
            "limitprice": "9.00000",
            "misc": "",
            "oflags": "fcib",
            "opentm": "0.000000",
            "refid": "OKIVMP-5GVZN-Z2D2UA",
            "starttm": "0.000000",
        "status": "open",
        "stopprice": "0.000000",
        "userref": 0,
        "vol": "0.00001000",
        "vol_exec": "0.00000000"
      }
    }
  ],
  "openOrders",
  {
    "sequence": 234
  }
]
```

Example of status-change payload

```
[
  [
    {
      "OGTT3Y-C6I3P-XRI6HX": {
        "status": "closed"
      }
    },
    {
      "OGTT3Y-C6I3P-XRI6HX": {
        "status": "closed"
      }
    }
  ],
  "openOrders",
  {
    "sequence": 59342
  }
]
```

addOrder

Request. Add new order.

Payload

Name	Type	Description
event	string	addOrder
token	string	Session token string
reqid	integer	Optional - client originated requestID sent as acknowledgment in the message response
ordertype	string	Order type - market|limit|stop-loss|take-profit|trailing-stop|stop-loss-limit|take-profit-limit|settle-position|trailing-stop-limit

Name	Type	Description
type	string	Side, buy or sell
pair	string	Currency pair
price	decimal	Optional dependent on order type - order price
price2	decimal	Optional dependent on order type - order secondary price
volume	decimal	Order volume in base currency
leverage	integer	amount of leverage desired (optional; default = none)
reduce_only	boolean	If true, order will only reduce a currently open position, not increase it or open a new position (optional; default = false)
oflags	string	Optional - comma delimited list of order flags. viqc = volume in quote currency (not currently available), fcib = prefer fee in base currency, fciq = prefer fee in quote currency, nompp = no market price protection, post = post only order (available when ordertype = limit)
starttm	string	Optional - scheduled start time. 0 = now (default) +<n> = schedule start time <n> seconds from now <n> = unix timestamp of start time
expiretm	string	Optional - expiration time. 0 = no expiration (default) +<n> = expire <n> seconds from now <n> = unix timestamp of expiration time
deadline	string	Optional - RFC3339 timestamp (e.g. 2021-04-01T00:18:45Z). Range of valid offsets from now: 500 milliseconds to 60 seconds, default is 5 seconds. The engine will prevent this order from matching after this time, it provides protection against latency on time sensitive orders.
userref	string	Optional - user reference ID (should be an integer in quotes)
validate	string	Optional - validate inputs only; do not submit order
close[ordertype]	string	Optional - close order type.
close[price]	decimal	Optional - close order price.
close[price2]	decimal	Optional - close order secondary price.
timeinforce	string	Optional - time in force. Supported values include GTC (good-til-cancelled; default), IOC (immediate-or-cancel), GTD (good-til-date; expiretm must be specified).

Example of payload

```
{
  "event": "addOrder",
  "ordertype": "limit",
  "pair": "XBT/USD",
  "price": "9000",
  "token": "0000000000000000000000000000000000000000",
  "type": "buy",
  "volume": "10.123"
}
```

Example of payload when conditional close order is sent

```
{
  "close[ordertype]": "limit",
  "close[price]": "9100",
  "event": "addOrder",
  "ordertype": "limit",
  "pair": "XBT/USD",
  "price": "9000",
  "token": "0000000000000000000000000000000000000000",
  "type": "buy",
  "volume": "10"
}
```

Response payload

Name	Type	Description
event	string	addOrderStatus
reqid	integer	Optional - client originated requestID sent as acknowledgment in the message response
status	string	Status. "ok" or "error"
txid	string	order ID (if successful)
descr	string	order description info (if successful)
errorMessage	string	error message (if unsuccessful)

Example of payload

```
{
  "descr": "buy 0.01770000 XBTUSD @ limit 4000",
  "event": "addOrderStatus",
  "status": "ok",
  "txid": "ONPNXH-KMKMU-F4MR5V"
}
```

```
{
  "errorMessage": "EOrder:Order minimum not met",
  "event": "addOrderStatus",
  "status": "error"
}
```

editOrder

Request. Edit open order

Payload

Name	Type	Description
event	string	editOrder
token	string	Session token string
orderid	string	Original Order ID or userref.
reqid	integer	Optional - client originated requestID sent as acknowledgment in the message response
pair	string	Currency pair
price	decimal	Optional dependent on order type - order price
price2	decimal	Optional dependent on order type - order secondary price
volume	decimal	Order volume in base currency
oflags	string	Optional - comma delimited list of order flags. post = post only order (available when ordertype = limit)
newuserref	string	Optional - user reference ID for new order (should be an integer in quotes)
validate	string	Optional - validate inputs only; do not submit order

Example of payload

```
{
  "event": "editOrder",
  "newuserref": "666",
  "oflags": "",
  "orderid": "O26VH7-COEPR-YFYXLK",
  "pair": "XBT/USD",
  "price": "9000",
  "reqid": 3,
  "token": "0000000000000000000000000000000000000000"
}
```

Response payload

Name	Type	Description
event	string	editOrderStatus
txid	string	order ID (if successful)
originaltxid	string	order ID (if successful)
reqid	integer	Optional - client originated requestID sent as acknowledgment in the message response
status	string	Status. "ok" or "error"

Name	Type	Description
descr	string	order description info (if successful)
errorMessage	string	error message (if unsuccessful)

Example of payload

```
{
  "descr": "order edited price = 9000.00000000",
  "event": "editOrderStatus",
  "originaltxid": "O65KZW-J4AW3-VFS74A",
  "reqid": 3,
  "status": "ok",
  "txid": "OTI672-HJFAO-XOIPPK"
}
```

cancelOrder

Request. Cancel order or list of orders.

For every cancelOrder message, an update message 'cancelOrderStatus' is sent. For multiple orderid in cancelOrder, multiple update messages for 'cancelOrderStatus' will be sent.

For example, if a cancelOrder request is sent for cancelling three orders [A, B, C], then if two update messages for 'cancelOrderStatus' are received along with an error such as 'EOrder: Unknown order', then it would imply that the third order is not cancelled. The error message could be different based on the condition which was not met by the 'cancelOrder' request.

Payload

Name	Type	Description
event	string	cancelOrder
token	string	Session token string
reqid	integer	Optional - client originated requestID sent as acknowledgment in the message response
txid	array	Array of order IDs to be canceled. These can be user reference IDs.

Example of payload

```
{
  "event": "cancelOrder",
  "token": "0000000000000000000000000000000000000000",
  "txid": [
    "OGTT3Y-C6I3P-XRI6HX",
    "OGTT3Y-C6I3P-X2I6HX"
  ]
}
```

Response payload

Name	Type	Description
event	string	cancelOrderStatus
reqid	integer	Optional - client originated requestID sent as acknowledgment in the message response
status	string	Status. "ok" or "error"
errorMessage	string	error message (if unsuccessful)

Example of payload

```
{
  "event": "cancelOrderStatus",
  "status": "ok"
}
```

```
{
  "errorMessage": "EOrder:Unknown order",
  "event": "cancelOrderStatus",
  "status": "error"
}
```

cancelAll

Request. Cancel all open orders. Includes partially-filled orders.

Payload

Name	Type	Description
event	string	cancelAll
token	string	Session token string
reqid	integer	Optional - client originated requestID sent as acknowledgment in the message response

Example of payload

```
{
  "event": "cancelAll",
  "token": "0000000000000000000000000000000000000000"
}
```

Response payload

Name	Type	Description
event	string	cancelAllStatus
reqid	integer	Optional - client originated requestID sent as acknowledgment in the message response
count	integer	Number of orders cancelled.
status	string	Status. "ok" or "error"
errorMessage	string	error message (if unsuccessful)

Example of payload

```
{
  "count": 2,
  "event": "cancelAllStatus",
  "status": "ok"
}
```

cancelAllOrdersAfter

Request.

cancelAllOrdersAfter provides a "Dead Man's Switch" mechanism to protect the client from network malfunction, extreme latency or unexpected matching engine downtime. The client can send a request with a timeout (in seconds), that will start a countdown timer which will cancel *all* client orders when the timer expires. The client has to keep sending new requests to push back the trigger time, or deactivate the mechanism by specifying a timeout of 0. If the timer expires, all orders are cancelled and then the timer remains disabled until the client provides a new (non-zero) timeout.

The recommended use is to make a call every 15 to 30 seconds, providing a timeout of 60 seconds. This allows the client to keep the orders in place in case of a brief disconnection or transient delay, while keeping them safe in case of a network breakdown. It is also recommended to disable the timer ahead of regularly scheduled trading engine maintenance (if the timer is enabled, all orders will be cancelled when the trading engine comes back from downtime - planned or otherwise).

Payload

Name	Type	Description
event	string	cancelAllOrdersAfter
token	string	Session token string
reqid	integer	Optional - client originated requestID sent as acknowledgment in the message response
timeout	integer	Timeout specified in seconds. 0 to disable the timer.

Example of payload

```
{
  "event": "cancelAllOrdersAfter",
  "reqid": 1608543428050,
  "timeout": 60,
  "token": "0000000000000000000000000000000000000000"
}
```

```
{
  "event": "cancelAllOrdersAfter",
  "reqid": 1608543428051,
  "timeout": 0,
  "token": "0000000000000000000000000000000000000000"
}
```

Response payload

Name	**Type**	**Description**
event	string	cancelAllOrdersAfterStatus
reqid	integer	Optional - client originated requestID sent as acknowledgment in the message response
status	string	Status. "ok" or "error"
currentTime	string	Timestamp (RFC3339) reflecting when the request has been handled (second precision, rounded up)
triggerTime	string	Timestamp (RFC3339) reflecting the time at which all open orders will be cancelled, unless the timer is extended or disabled (second precision, rounded up)
errorMessage	string	error message (if unsuccessful)

Example of payload

```
{
  "currentTime": "2020-12-21T09:37:09Z",
  "event": "cancelAllOrdersAfterStatus",
  "reqid": 1608543428050,
  "status": "ok",
  "triggerTime": "2020-12-21T09:38:09Z"
}
```

```
{
  "currentTime": "2020-12-21T09:37:09Z",
  "event": "cancelAllOrdersAfterStatus",
  "reqid": 1608543428051,
  "status": "ok",
  "triggerTime": "0"
}
```